PolyPublie
Polytechnique Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Titre: Title:	Rétroaction de la qualité de l'expérience pour améliorer la qualité de service
Auteur: Author:	Gregory Charlot
Date:	2012
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Charlot, G. (2012). Rétroaction de la qualité de l'expérience pour améliorer la qualité de service [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/957/

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/957/
Directeurs de recherche: Advisors:	Samuel Pierre
Programme: Program:	Génie informatique

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal

https://publications.polymtl.ca

UNIVERSITÉ DE MONTRÉAL

RÉTROACTION DE LA QUALITÉ DE L'EXPÉRIENCE POUR AMÉLIORER LA QUALITÉ DE SERVICE

GREGORY CHARLOT
DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION DU DIPLÔME DE
MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
NOVEMBRE 2012

© Gregory Charlot, 2012.

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

RÉTROACTION DE LA QUALITÉ DE L'EXPÉRIENCE POUR AMÉLIORER LA QUALITÉ DE SERVICE

présenté par : CHARLOT Gregory
en vue de l'obtention du diplôme de : Maîtrise ès Sciences Appliquées
a été dûment accepté par le jury d'examen constitué de :

M. GAGNON Michel, Ph.D, président
M. PIERRE Samuel, Ph.D., membre et directeur de recherche
M. QUINTERO Alejandro, Doct., membre

À mes parents, Ronald et à toute ma famille.

REMERCIEMENTS

J'éprouve un immense plaisir d'exprimer ma gratitude envers Dieu, Jehovah, et envers toutes les personnes qui m'ont aidé à la réalisation de ce mémoire.

Je veux d'abord remercier mon directeur de recherche, le Professeur Samuel Pierre, pour son encadrement, son support financier et ses conseils nécessaires à la réalisation de ce travail.

Mes remerciements s'adressent également à Méral Shirazipour chargée de mon encadrement chez Ericsson pour m'avoir guidé, orienté et soutenu dans toutes les étapes de ce projet de recherche.

Je tiens aussi à remercier les honorables membres du jury qui ont accepté d'évaluer ce travail.

Enfin je veux remercier tous les collègues et amis du LAboratoire de recherche en Réseautique et Informatique mobile (LARIM) pour leur collaboration, particulièrement Marième Diallo, Saida Maaroufi El Idrissi et Valérie Justafort.

RÉSUMÉ

Pendant de nombreuses années l'évaluation des réseaux IP a été effectuée de façon objective en mesurant différents critères qui déterminent la qualité du réseau. Ces critères font référence à un ensemble de paramètres de qualité de service (QdS) que doit satisfaire le réseau dans le transport des paquets. Avec l'apparition des trafics multimédias tels que la voix sur IP et la vidéo, les paramètres de QdS, quoique fondamentaux, se révèlent être insuffisants puisqu'ils ne tiennent pas compte de ce qui se passe en amont et en aval du réseau. C'est ce qui explique l'émergence du concept de la qualité de l'expérience (QdE) dans les réseaux IP pour les applications temps réel et interactives. Ce concept, exprimant le niveau de satisfaction de l'usager, permet de tenir compte de ce dernier dans la boucle de service. Ainsi, il s'avère nécessaire de disposer de mécanismes pour contrôler la QdE des usagers et qui sont capables de réagir lorsqu'une dégradation se produit.

Les mécanismes proposés jusqu'ici concernent, pour la plupart, les applications de voix sur IP ou la vidéo. Ils agissent le plus souvent au niveau des équipements terminaux particulièrement sur les codeurs ou les décodeurs en modifiant leur débit ou en faisant varier la taille du tampon de la gigue. De ce fait, ils ne sont appropriés que pour les applications multimédias et ne peuvent pas être appliqués aux autres applications interactives, les jeux vidéo en ligne par exemple, auxquelles le concept de QdE s'est élargi. De plus, très peu de travaux proposent d'utiliser les métriques de QdE pour avoir de meilleurs mécanismes de QdS.

Dans ce mémoire, un nouveau mécanisme est proposé qui, via des techniques utilisées pour faire de la QdS dans les réseaux IP, permet de réagir lorsque la QdE, pour une application, subit des dégradations. Ce mécanisme comporte deux aspects. Le premier aspect consiste à définir un cadre réaliste pour, d'une part, avoir la rétroaction de la QdE (*feed-back*) pour tout type d'applications, temps réel et non temps réel ; et d'autre part, introduire le *feed-back* de la QdE (*re-feed-back*) dans le réseau de sorte que les différents nœuds puissent en être informés. La réinsertion du *feed-back* de la QdE se fait '*in band*' à travers l'entête IP. Le second aspect consiste à proposer un nouveau mécanisme de classification des paquets par les routeurs internes à un domaine DiffServ. Ce mécanisme permet aux routeurs de tenir compte de la QdE lors de la mise en file d'attente des paquets d'un flot de trafic qui doit bénéficier d'une certaine garantie de QdE. L'information de la QdE sera accessible aux routeurs par le mécanisme de *feed-back* proposé qui l'écrira dans l'entête des paquets IP.

Le mécanisme proposé a été simulé avec le simulateur NS-3. Les résultats obtenus dé-

montrent que l'utilisation du *feed-back* de la métrique de la QdE dans DiffServ permet d'obtenir de meilleurs résultats de QdE, prouvant ainsi qu'il est possible et utile d'impliquer la QdE dans les mécanismes de QdS.

ABSTRACT

For many years the evaluation of IP networks was carried out objectively by measuring different criteria that determine the quality of the network. Those criteria refer to a set of parameters of quality of service (QoS) that must be satisfied by the network while carrying packets. With the emergence of multimedia traffic such as Voice over IP (VoIP) and video, QoS parameters, though still very important, are yet not enough because they do not take into account what is happening at the end nodes. This has led to the emergence of the new concept of quality of experience (QoE) which expresses the user's satisfaction regarding a given comunication service. Thus it is necessary to develop mechanisms in IP netwoks allowing one to monitor and enhance the users' QoE.

So far, proposed mechanisms concern mostly VoIP or video applications and act, most of the time, at the terminal equipment, particularly at the codec to adapt, if possible, the data transmission rate or to modify the jitter buffer size. So, they are only suitable for multimedia applications and they do not take into account other interactive applications, like online games for example, to which QoE concept has widened. In addition, very few works propose to use QoE metrics to enhance QoS mechanisms.

In this master thesis, we propose a new QoS-based mechanism which is able to react when the QoE value of an application goes below a threshold. This mechanism has two aspects. The first aspect is to define a realistic framework to have the QoE information echoed to the receiver on the one hand and secondly to insert the QoE information in the network so that nodes on the path can use it in QoS mechanisms. That is done 'in band' through IP packet header. The second aspect consist of proposing a new mechanism to classify packets by routers inside a DiffServ domain. This latter mechanism enables routers to take into account the QoE info. The QoE information will be accessible to the routers by the proposed feedback mechanism.

The proposed mechanism was simulated with the NS-3 simulator. The results show that the use of feedback of the QoE info in DiffServ achieves better QoE results, proving that it is possible and useful to involve the QoE in QoS mechanisms.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES ET ABRÉVIATIONS

AF	Assured Forwarding
AMR	Adaptive Multi-Rate
ARED	Adaptative Random Early Detection
AQM	Active Queue Management
AVQ	Adaptative Virtual Queue
BBRR	bit-by-bit Round-Robin
CBR	Constant Bit Rate
DF	Default Forwarding
DiffServ	Differentiated Services
DSCP	DiffServ Code Point
DWRR	Deficit Weighted Round Robin
ECN	Explicit Congestion Notification
EF	Expedited Forwarding
ETSI	European Telecommunications Standards Institute
FIFO	First In First Out
FQ	Fair Queueing
FTP	File Transfer Protocol
GSM	Global System for Mobile Communications
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IntServ	Integrated Services
IP	Internet Protocol
IPTV	Internet Protocol television
IPv4	IP version 4
IPv6	IP version 6
ITU	International Telecommunication Union
MOS	Mean Opinion Score
NS-2	Network Simulator version 2
NS-3	Network Simulator version 3
OSI	Open Systems Interconnection
PCAP	Packet Capture
PEAQ	Perceptual Evaluation of Audio Quality
PESQ	Perceptual Evaluation of Speech Quality

PHB	Per-Hop Behavior
PI	Proportional Integrator
PQ	Priority Queueing
POLQA	Perceptual Objective Listening Quality Assessment
PSQM	Perceptual Speech Quality Measure
QdS	Qualité de Service
QdE	Qualité de l'Expérience
RED	Random Early Detection
REM	Random Exponential Marking
RFC	Request for Comments
RR	Round-Robin
RTCP	Real-time Transport Control Protocol
RTP	Real Time Protocol
SLA	Service Level Agreement
srTCM	single-rate, three-color marker
TCP	Transmission Control Protocol
trTCM	two-rate, three-color marker
TSpec	Trafic Specification
RSVP	Resource Reservation Protocol
RSpec	Request Specification
RTT	Round Trip Time
UDP	User Datagram Protocol
VoIP	Voice over IP
WBBRR	Weighted bit-by-bit Round-Robin
WFQ	Weighted Fair Queueing
WRED	Weighted Random Early Detection
WRR	Weighted Round-Robin

CHAPITRE 1

INTRODUCTION

Au cours des vingt dernières années, la réseautique a connu une évolution importante. Nous avons assisté, depuis l'avènement du World Wide Web vers les années 1990, à la naissance de nouveaux types de trafic et à une augmentation considérable des usagers de l'internet. Cette évolution a fait naître le besoin d'avoir une gestion efficace des ressources pour assurer que le réseau puisse acheminer correctement le trafic des usagers par la mise en place de politiques de qualité de service (QdS). De plus, les progrès technologiques et le besoin d'avoir une connexion permanente ont contribué à réaliser le passage des réseaux filaires aux réseaux sans fil et mobiles. L'émergence des technologies mobiles et l'intérêt grandissant des usagers pour les communications multimédias font que ces derniers deviennent de plus en plus intéressés par la qualité qu'ils perçoivent et non par les paramètres techniques relatives à la transmission des trafics.

En téléphonie classique, la qualité perçue pas les usagers est un élément fondamental, car une communication téléphonique effectuée dans les mêmes conditions peut être perçue différemment par deux couples de correspondants différents. L'un pourrait être très satisfait de la qualité de la communication et l'autre plus ou moins satisfait. C'est la raison pour laquelle les opérateurs utilisaient le *Mean Opinion Score* (MOS) [1] pour évaluer la qualité des appels téléphoniques telle que perçue par les usagers. Il s'agissait de recueillir les opinions de différents usagers sélectionnés suivant une méthodologie. Chaque opinion est exprimée par une note sur une échelle de 1 à 5 et la moyenne des notes obtenues indiquait le niveau de satisfaction des usagers. Ce niveau de satisfaction est désigné par le nouveau concept de qualité d'expérience (QdE) [2].

Tableau 1.1 *Mean Opinion Score - MOS*

MOS	Qualité	Défaut de la communication
5	Excellent	Imperceptible
4	Bon	Perceptible, pas nuisible
3	Satisfaisant	Légèrement nuisible
2	Médiocre	Nuisible
1	Mauvais	Très nuisible

Avec la convergence des réseaux et des services où l'infrastructure de l'internet est utilisée

pour transporter tout type de trafic, un intérêt sans cesse croissant se porte sur la QdE dans les réseaux IP si bien que son application s'est étendu au-delà de la téléphonie classique. Elle concerne aujourd'hui tous les services interactifs accessibles via un réseau IP tels que la voix sur IP (VoIP), la télévision sur IP (IPTV), le web et les jeux vidéo en ligne. La QdE devient un élément critique pour un fournisseur de service qui veut satisfaire ses clients et ainsi rester dans la compétition [3]. D'où la nécessité, pour les opérateurs, de s'assurer de la fidélité de leurs utilisateurs, directement liée à leur satisfaction, pendant qu'ils cherchent à maximiser leur profit en optimisant leurs ressources. A cet effet, les opérateurs doivent disposer, dans leurs réseaux, des mécanismes pour constamment contrôler la QdE et réagir dès qu'une dégradation se produit.

Dans la suite de ce chapitre nous décrirons brièvement les concepts de QdS et de QdE tout en faisant ressortir leurs différences. Ensuite certains éléments de problématique relatifs au concept de QdE seront exposés à la suite desquels les objectifs du mémoire seront énoncés. Enfin le plan du mémoire sera fourni.

1.1 Définitions et concepts de base

Qualité de service/Qualité de l'expérience

Il existe différentes définitions pour le concept de QdS. Dans [4], Crawley la définit comme un ensemble de requis que doit satisfaire un réseau en transportant un trafic donné. Ces requis sont traduits par des métriques qui font référence à la capacité du réseau à transporter les informations en terme de disponibilité, au délai maximal que doit expérimenter les paquets, au taux de perte de paquets minimal que doit garantir le réseau, etc. [5]. Pour assurer que ces métriques aient des valeurs permettant de satisfaire les requis, différents stratégies ou mécanismes sont utilisés par les opérateurs et administrateurs de réseaux. Ces techniques incluent le routage, la régulation de trafic, l'ordonnancement de paquets, le contrôle du trafic, le contrôle d'admission, etc.

La QdE désigne, suivant la définition de l'*European Telecommunications Standards Institute* (ETSI), une mesure de performance du point de vue de l'usager d'un système de communication basée sur des mesures psychologiques objectives et subjectives [3]. Elle prend en compte des paramètres techniques tels que les paramètres de QdS et est influencée par le contexte dans lequel la communication se fait et les attentes de l'usager. Elle est une mesure importante de la performance globale d'un système de communication qui prend en compte ce qui se passe en amont et en aval du réseau jusqu'aux terminaux des utilisateurs (Figure 1.1).

Pour éviter toute confusion entre les concepts de QdS et de QdE, il importe de bien saisir

leur différence. Considérant le modèle de référence, OSI, la QdS concerne particulièrement les couches inférieures jusqu'à la couche transport et la QdE les couches supérieures (Figure 1.2).

La QdE revêt une importance capitale pour les fournisseurs de services et les opérateurs et est utile pour optimiser les services ou produits. Un fournisseur de services dont les clients expérimentent une bonne QdE jouit d'un avantage compétitif significatif tandis que d'autres qui n'y mettent pas assez l'accent pourront connaître des pertes de revenue ils se verront perdre des clients.

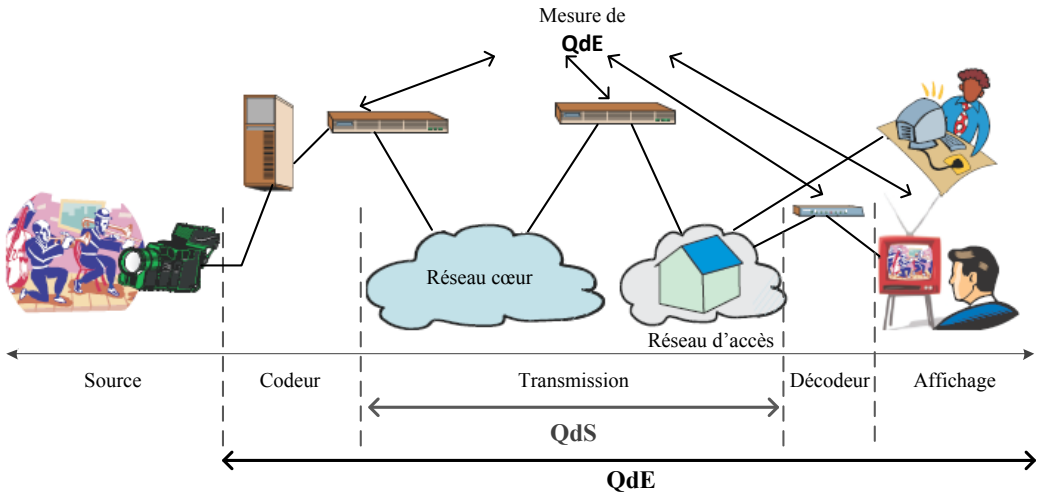


Figure 1.1 QdS/QdE domaines d'application inspiré de [6]

Évaluation de la QdE

En tant que mesure de performance d'un réseau et système de communication, d'après l'ETSI, différentes méthodes existent pour évaluer la QdE. Ces méthodes dépendent du type de l'application et sont catégorisées en méthodes subjectives, objectives et modèles de planification réseau. Les méthodes subjectives se basent sur l'opinion des usagers sur un service multimédia. Suivant ces méthodes, il est demandé aux usagers d'indiquer par des notes variant de 1 à 5 leur niveau de satisfaction et la moyenne donne le MOS. Quoiqu'elles soient considérées comme les méthodes les plus fiables pour évaluer la QdE, elles comportent, néanmoins, certains inconvénients. En effet, ces méthodes sont très fastidieuses, consomment beaucoup de temps et ne peuvent être utilisées en temps réel. Pour pallier aux limitations des méthodes subjectives, des méthodes objectives ont été élaborées et standardisées. Elles permettent d'évaluer la QdE en se basant sur un ensemble de paramètres liés à un service particulier ou sur des paramètres du signal à la sortie pour estimer la QdE. Ces méthodes sont

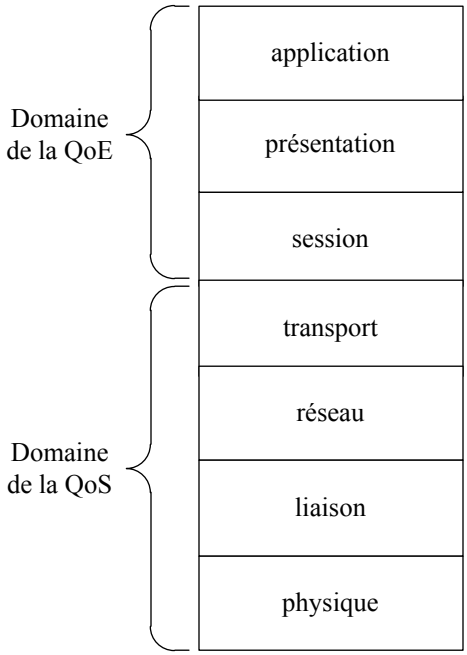


Figure 1.2 QdS/QdE couches dans le modèle OSI

subdivisées en quatre classes : *Full Reference* (FR), *No Reference* (NR), *Reduced Reference* (RR) et modèle de planification réseau.

La catégorie FR regroupe les méthodes qui comparent le signal à la source, considéré comme une référence, au signal déformé à la sortie pour donner une estimation de la QdE. Dans certains cas, les caractéristiques du signal d'entrée peuvent être inconnues. A cet effet les méthodes de la catégorie NR sont utilisées. Ces méthodes évaluent le signal à la sortie et une estimation de la QdE est déduite. Les méthodes de la classe RR, en revanche, font un compromis entre les deux autres catégories. Elles analysent le signal à la sortie à la lumière d'un ensemble réduit d'information sur le signal de référence. Les modèles de planification réseau sont considérés comme une sous-catégorie des méthodes objectives car ils ne se basent pas sur les opinions des usagers. Cependant, ils ne procèdent pas à une analyse du signal réel mais ils utilisent une fonction qui fait le lien entre la QdE et des paramètres intrinsèques du réseau notamment ceux de la QdS. Un exemple pour cette catégorie est le modèle E (*E-model*) [7].

Le Tableau 1.2 liste différentes méthodes pour évaluer la QdE.

Tableau 1.2 Méthode d'évaluation de la QdE

VoIP	VIDEO	WEB	Jeux en ligne
PSQM, ITU-T Rec. P.861 (08/1996)	ITU-T Rec. J.249	ITU-T Rec. G.1030	OPScore (Online Playability Score) proposé par Ubicom
PESQ, ITU-T Rec. P.862 (02/2001)	ITU-T Rec. J.146		G-Model
POLQA, ITU-T Rec. P.862 (01/2011)	TU-T Rec. J.144		
PEAQ, ITU-R Rec. BS.1387	ITU-R BT.1683		
E-Model, ITU-T Rec. G.107 (03/2005)	ITU-T Rec. J.247		

1.2 Éléments de la problématique

Comme mentionné ci-dessus, l'objectif de tout fournisseur de services ou opérateur est de satisfaire leur client ou de leur permettre d'expérimenter une bonne qualité d'expérience tout en optimisant les ressources du réseau. Étant donné que la QdE peut être considérée comme une métrique de performance globale d'un système de communication, son évaluation inclut celle de la QdS. De ce fait, différentes propositions ont été faites en vue de trouver une relation mathématique entre la QdE et la QdS. À travers ces propositions certains chercheurs se sont donnés pour objectif de définir des relations mathématiques telles que $QdE = f(QdS)$ [8] dans l'objectif de contrôler la QdE par les paramètres de QdS. D'autres travaux portent sur des mécanismes qui, en utilisant la rétroaction de la QdE évaluée au récepteur, permettent d'agir sur le débit des codeurs utilisés dans le cas des applications de voix ou de vidéo lorsque la QdE de l'usager se dégrade [9]. Cette catégorie de travaux vise essentiellement à améliorer la QdE dans le cas d'une communication de voix sur IP et ne saurait être utilisée pour d'autres types d'applications.

D'autres auteurs proposent de nouveaux mécanismes de QdS ou des modifications à apporter à des mécanismes existants pour pouvoir améliorer la QdE. Dans cette troisième catégorie, certains se portent sur de nouveaux mécanismes de gestion des files d'attentes, l'ordonnancement de paquets et d'autres sur le routage [10]. La plupart des mécanismes proposés ne réagissent pas lorsque la QdE de l'usager se dégrade, et ceux qui le font, ne mettent l'accent que sur les applications multimédias où il est possible de faire varier le débit des codeurs et décodeurs ; ce qui n'est pas valable pour les autres types d'application.

1.3 Objectifs de la recherche

L'objectif de ce mémoire est de proposer un mécanisme qui permet à un réseau IP de réagir lorsque la QdE, pour certaines applications, subit des dégradations, en utilisant l'architecture de QdS *Differentiated Services* (DiffServ), dans le but d'assurer que le réseau garantisse un niveau acceptable de QdE pour ces applications. Les autres types de trafic ne doivent pas être pénalisés et l'usage des ressources du réseau doit être optimisé. De manière plus spécifique le mémoire vise à :

1. Analyser les solutions déjà implémentées ou proposées dans la littérature relatives à l'amélioration de la QdS dans les réseaux IP ;
2. Examiner les propositions existantes dans la littérature qui mettent en évidence l'impact de la QdS sur la QdE et vice-versa ;
3. Proposer un nouveau mécanisme réaliste qui permet d'insérer l'information de la QdE dans un réseau pour être utilisée dans des mécanismes de QdS existants ;
4. Proposer une modification dans DiffServ pour prendre en compte l'information de la QdE ;
5. Évaluer la performance de la proposition au moyen de simulations.

1.4 Plan du mémoire

Les objectifs 1 et 2 seront atteints à travers la revue de littérature qui fera l'objet du chapitre 2 où nous présenterons différentes techniques qui ont été proposées pour fournir la QdS dans les réseaux IP et montrerons, à travers certains travaux, l'impact sur la QdE de la QdS et vice versa. Après avoir fait ressortir les limites des solutions implémentées ou proposées relatives à l'amélioration de la QdE dans les réseaux IP, la proposition sera élaborée et les mécanismes seront décrits dans le chapitre 2. Au chapitre 3, sera décrit le simulateur utilisé pour évaluer la performance des mécanismes proposés et l'analyse des résultats sera effectuée. Enfin, la conclusion fera la synthèse. Nous y ferons ressortir les limitations des travaux et y évoquerons également certains pistes pour des travaux futurs.

CHAPITRE 2

DÉFIS DE QUALITÉ DE SERVICE ET IMPACT SUR LA QUALITÉ DE L'EXPÉRIENCE

Les réseaux IP sont par essence de type *best-effort* où tous les trafics sont traités de la même façon sans tenir compte de leurs besoins particuliers. Avec l'évolution de l'internet, il a paru nécessaire d'adjoindre au modèle *best-effort*, des techniques qui permettent d'utiliser plus efficacement les ressources réseaux par une gestion pro-active de la congestion, d'assurer une utilisation équitable des ressources par les différents flots de trafic et enfin de différencier les trafics en leur accordant des priorités pour répondre à leurs besoins spécifiques de qualité de service (QdS).

Dans ce chapitre seront présentés, dans un premier temps, différents stratégies et modèles de QdS qui ont été implémentés en vue d'améliorer la QdS dans les réseaux IP. Dans un second temps, seront passés en revue quelques propositions et travaux qui montrent l'impact de la QdS sur la QdE.

2.1 Stratégies et mécanismes élémentaires de qualité de service

Les stratégies élémentaires de QdS incluent : les mécanismes de gestion pro-active des files d'attente des routeurs qui aident à réduire la congestion, les algorithmes d'ordonnancement, les techniques de contrôle de trafic etc. Bien qu'elles soient utilisées dans certains cas de façon individuelle, elles constituent les modules sur lesquels se fondent les principales architectures de QdS.

2.1.1 Gestion active des files d'attente - AQM

Les mécanismes de gestion active des files d'attente (*active queue management*) sont des mécanismes de gestion pro-active des tampons des routeurs pour éviter que ces derniers soient congestionnés. En effet, les routeurs disposent de mémoires tampons qui absorbent les paquets qui ne peuvent être transmis immédiatement autrement dit les paquets sont mis dans une file d'attente. Traditionnellement, ces files d'attente sont gérées suivant le mécanisme *Drop-Tail* illustré à la Figure 2.1(a) qui rejette les paquets lorsque la mémoire tampon du routeur est remplie. Bien que cette méthode soit la plus simple à mettre en œuvre, elle a néanmoins

plusieurs inconvénients. Elle est inéquitable en terme d'allocation de ressources aux différents flots de trafic qui transitent à travers les routeurs et peut entrainer des pertes de paquets en rafale en période de congestion. Les sources produisant des rafales importantes risquent d'être beaucoup plus pénalisées que celles à débit constant et il est possible de se retrouver dans une situation où des files d'attente sont remplies en permanence, ce qui se traduira par une augmentation du délai expérimenté par les paquets.

Les mécanismes d'*AQM* représentent une alternative pour éviter ou diminuer la congestion dans le réseau, réduire le taux d'occupation des files d'attente en assurant une bonne utilisation des liens et une distribution équitable des ressources.

L'algorithme *Red Early Drop* (RED), illustré à la Figure 2.1 (b) et proposé par Sally Floyd [11] est l'algorithme d'AQM le mieux connu. Il utilise l'approche de la moyenne mobile exponentielle pour calculer la taille moyenne de la file Q_{moy}. Cette moyenne est comparée à deux seuils définis pour la taille de la file, un seuil minimal th_{min} et un seuil maximal th_{max}. Lorsqu'un paquet arrive, l'algorithme compare la taille moyenne de la file d'attente à ces deux seuils :

- Si la taille moyenne de la file Q_{moy} est inférieure au seuil minimal, th_{min}, le paquet est inséré dans la file d'attente.
- Si la taille moyenne de la file Q_{moy} est supérieure au seuil minimal, th_{min} mais inférieure au seuil maximal, th_{max}, ce qui indique qu'il y a un début de congestion et le paquet est rejeté avec une probabilité p variant linéairement entre 0 et P_{max}.
- Si la taille moyenne de la file Q_{moy} est supérieure th_{max}, il dénote une congestion persistante et le paquet est rejeté pour éviter que la file ne soit pas remplie de façon persistante.

2.1.2 Ordonnancement de paquet

L'ordonnancement des paquets est le mécanisme par lequel le routeur décide du prochain paquet à envoyer sur le lien. Quand plusieurs files d'attente existent dans le routeur, l'ordonnancement permet de choisir la prochaine file d'attente qui doit être servie. L'algorithme d'ordonnancement de base et le plus utilisé est l'algorithme premier arrivé, premier servi (FIFO) où les paquets sont servis dans l'ordre de leur arrivée. Il n'établit pas de distinctions entre les paquets de classes de trafic différentes ni ne contrôle le partage des ressources. En conséquence, un flot peut monopoliser toute la capacité d'un routeur dans une période où ses paquets arrivent en rafale. Pour pallier à certains des inconvénients de l'algorithme FIFO notamment en ce qui a trait au contrôle du partage des ressources, la prise en charge de plusieurs classes de service ou la réduction du temps d'attente dans les files d'attente, deux

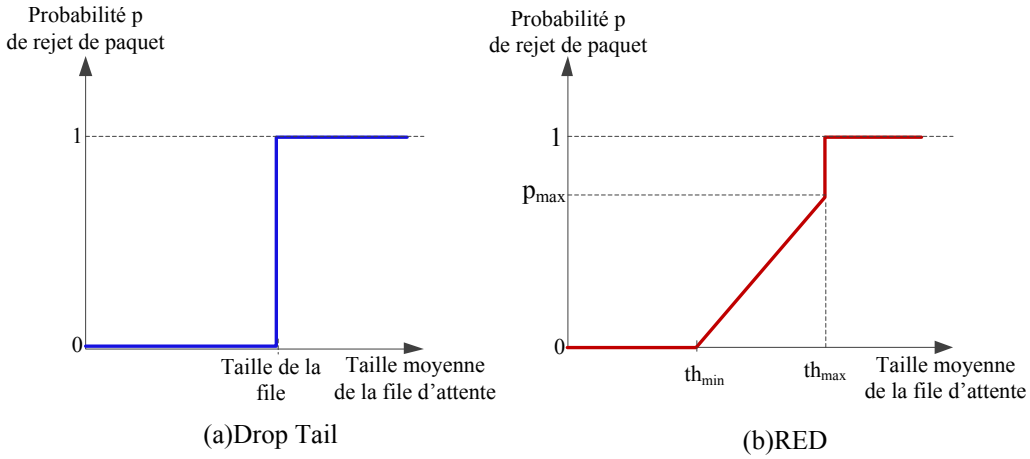


Figure 2.1 Gestion des files d'attente *Drop Tail* et RED

autres types d'algorithmes ont été proposés : les algorithmes avec priorité (*priority queuing*) et les algorithmes de partage de la bande passante.

Parmi les algorithmes avec priorité, l'algorithme *priority queuing* (PQ) est le plus populaire. Dans ce dernier, les paquets sont d'abord classifiés par le système avant d'être mis dans des files d'attente auxquelles des priorités sont accordées. Les différentes files d'attente sont servies dans l'ordre de leur priorité. La file la plus prioritaire est servie en premier et tant qu'elle contient des paquets aucune autre file n'est servie. À l'intérieur de chaque file l'ordonnancement se fait suivant l'algorithme FIFO. L'avantage principal de l'ordonnancement avec priorité est la subdivision du trafic en différentes classes de façon à accorder un traitement préférentiel à certains types de trafic. Par exemple, les priorités peuvent être fixées de sorte que les applications temps réel reçoivent des priorités supérieures par rapport aux autres applications. Cependant, ce type d'algorithme a un défaut évident en présentant un risque de famine pour les files de basses priorités s'il y a constamment du trafic de classes de priorités élevées. Pour remédier à ce problème de famine, une variante de l'algorithme PQ, *rate-controlled priority queuing*, a été proposée afin de limiter le trafic des classes de plus grande priorité. D'autres mécanismes externes peuvent aussi être utilisés pour limiter les flux qui partagent les files les plus prioritaires.

Les algorithmes *Round Robin* (RR) et *Fair Queuing* (FQ) sont utilisés pour contrôler le partage de la bande passante. Se basant sur le principe de la répartition des paquets dans différentes files d'attente, ces algorithmes servent les files d'attente à tour de rôle. Ils parcourent en séquence les différentes files et prennent le paquet se trouvant à l'entête de chacune d'elle. Si une file est vide, ils passent à la suivante. Un problème d'équité peut se

produire si les paquets dans les différentes files ne sont pas de la même taille. Pour pallier à ce problème éventuel d'équité, diverses variantes ont été proposées. L'algorithme *Weighted Round Robin* (WRR) associe un poids à chacune des files d'attente. Lorsque l'ordonnanceur fait un tour, il sert dans chacune des files une quantité de paquets équivalent au poids de la file. Le poids d'une file se réfère à la quantité de bande passante allouée à cette file. Cet algorithme contrôle la quantité de bande passante allouée à chaque classe de trafic. Cependant, ce contrôle ne peut être efficace que si tous les paquets ont la même taille ou quand la taille moyenne des paquets est connue à l'avance.

L'algorithme *Deficit Weighted Round Robin* (DWRR) permet d'adresser les limitations de l'algorithme WRR en garantissant une répartition juste de la bande passante lorsqu'il sert des files contenant des paquets de tailles inégales. Dans cet algorithme, outre le poids (*weight*) qui indique la proportion de bande passante allouée à une file, chaque file est configurée avec deux autres paramètres :

- Un compteur de déficit (*DeficitCounter*) qui spécifie le nombre total d'octets que la file peut transmettre à chaque fois qu'elle est visitée par l'ordonnanceur. Le compteur de déficit permet à une file qui ne pouvait pouvait pas transmettre lors du passage précédent de l'ordonnanceur, parce que la taille du paquet qui occupait l'entête de la file était supérieure à la valeur de *DefictCounter*, d'avoir des crédits qu'elle peut utiliser lors du prochain passage de l'ordonnanceur.
- Un quantum de service qui est proportionnel au poids et qui est exprimé en octets. Le compteur de déficit pour une file est incrémenté par le quantum chaque fois que la file est visitée par l'ordonnanceur.

Typiquement, à chaque tour, l'ordonnanceur visite chacune des files non vide et détermine la taille en octet du paquet se trouvant à l'entête de la file. la variable *DeficitCounter* est incrémentée par la valeur du quantum. Si la taille du paquet se trouvant à l'entête de la file est supérieure à la variable *DefictCounter*, alors l'ordonnanceur passe à une autre file. Si elle est inférieure ou égale à la variable *DefictCounter*, alors le paquet est envoyé sur le lien. L'ordonnanceur continue à servir les paquets et à décrémenter la variable *DefictCounter* de la taille des paquets transmis jusqu'à ce que la taille du paquet qui arrive à l'entête de la file devienne inférieure à *DefictCounter*, ou si la file est vide. Si la file est vide, la valeur de *DefictCounter* est mise à zéro et l'ordonnanceur passe à une autre file non vide.

Plusieurs autres algorithmes ont été proposés, cependant nous nous gardons de les présenter. Pour plus de détails, le lecteur peut se référer à [12].

2.1.3 Routage

De façon générale, le routage consiste en la détermination d'un chemin pour acheminer les données d'une application d'une source à une destination. En référence à la QdS, le routage est utilisé pour trouver, non pas un chemin, mais le meilleur chemin pour transporter les données d'une application tout en assurant une bonne utilisation et un partage juste des ressources. En choisissant le meilleur chemin, l'algorithme de routage doit permettre de satisfaire les contraintes de l'application en tenant compte de l'état actuel des ressources disponibles dans le réseau. Par exemple, pour les applications interactives fonctionnant en mode temps réel comme la voix sur IP, sera choisi un chemin permettant d'avoir un faible délai et un faible débit tandis qu'un chemin offrant un haut débit et un long délai sera plus approprié pour effectuer du téléchargement.

Différents types d'algorithmes de routage sont proposés dans la littérature [13] pour faire la QdS. Ces algorithmes de routage peuvent être catégorisés en algorithmes dynamiques ou statiques, en ligne ou hors ligne. Les algorithmes statiques utilisent des informations qui ne changent pas dans le temps pour établir les chemins tandis que ceux qui sont statiques utilisent des informations liées à l'état du réseau comme la bande passante disponible, la capacité des liens, etc. D'un autre coté, dans les algorithmes de routage en ligne, les chemins sont établis sur demande. Ils considèrent chaque requête de 'chemin' séparément ce qui empêche le routage à nouveau des connections routées. Les algorithmes hors ligne, par contre, sont utilisés dans des connexions permanentes car de nouvelles routes ne peuvent être calculées.

2.1.4 Contrôle et lissage de trafic

Le contrôle et le lissage de trafic permettent d'assurer la régulation de trafic c'est-à-dire le rythme moyen auquel les données sont transmises ainsi que les rafales. Le lissage du trafic assure le transport des paquets de sorte que le débit ne dépasse pas un seuil maximal. Les paquets qui ne sont pas directement acheminés sont bufférisés pour être acheminés plus tard. Ce mécanisme nécessite une allocation suffisante de ressources au niveau des routeurs pour pouvoir mettre dans un tampon les paquets qui sont retardés, ce qui permet d'éviter des pertes de paquets et de prévenir la congestion. Le contrôle de trafic, de son coté, permet de contrôler un ou plusieurs flots de trafic agrégés, mais à la différence du lissage de trafic, le trafic excédentaire n'est pas bufférisé mais rejeté.

Le lissage de trafic peut être implémenté en utilisant l'algorithme seau à jeton (*token bucket*) [14], décrit à la Figure 2.2, avec un seau de taille B qui représente la taille de rafale permise et un débit R de remplissage en jeton du seau. Quand un paquet arrive, la taille du

paquet est comparée au nombre de jetons disponibles dans le seau. Si le nombre d'octets du seau équivalent au nombre de jetons qu'il contient est au moins égal à la taille du paquet, le paquet est transmis sans délai et le seau est décrémenté du nombre de jetons égal au nombre d'octets du paquet. S'il y a moins de jetons que d'octets dans le paquet, le paquet est retardé c'est-a-dire mis en file d'attente jusqu'à ce qu'il y ait assez de jetons dans le seau.

Le seau à jeton peut aussi être utilisé pour effectuer du contrôle de trafic. A la différence du lissage, lorsqu'un paquet arrive et que le nombre d'octets jetons est inférieur à la taille du paquet, le paquet est rejeté.

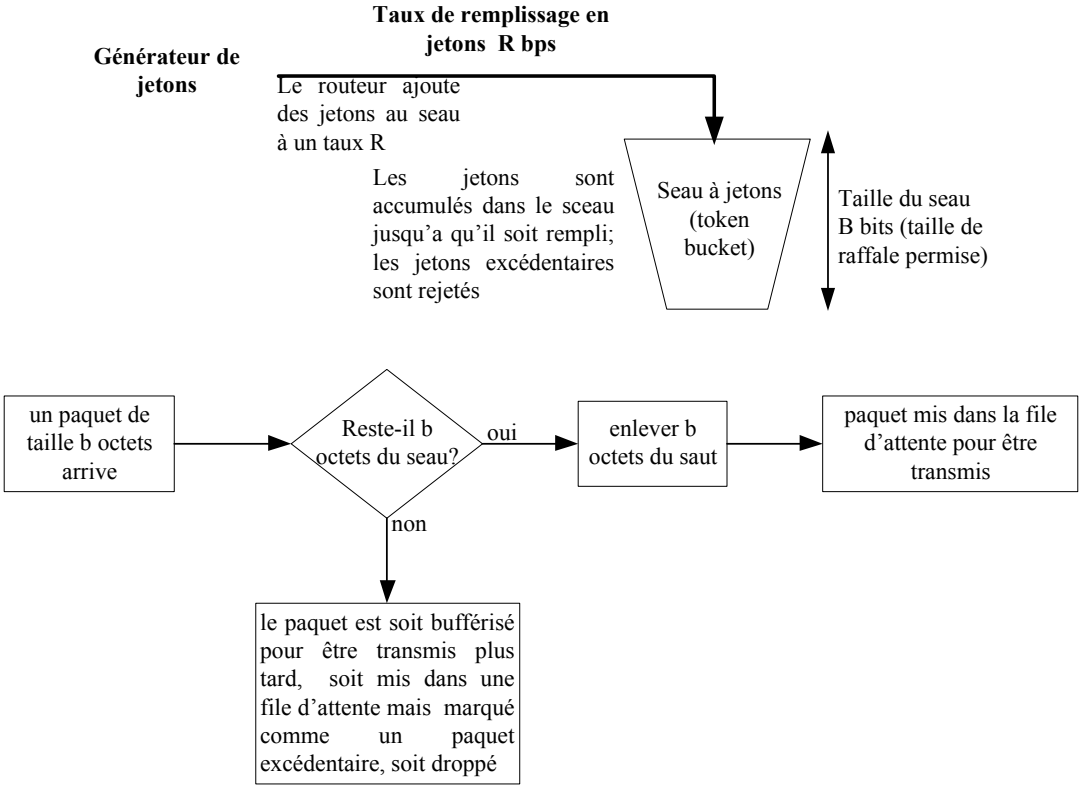


Figure 2.2 Algorithme Seau à Jetons

2.2 Modèles de QdS - Architectures de QdS dans les réseaux IP

Dans un réseau IP, les mécanismes décrits dans la précédente section peuvent être combinés pour constituer des architectures de QdS. Ces architectures permettent ainsi d'assurer une qualité de service de bout en bout répondant aux exigences des usagers définis dans

un contrat de service entre ces derniers et les fournisseurs de service. Deux architectures de qualité de service ont été standardisées par l'IETF

1. L'architecture des services différenciés DiffServ
2. L'architecture à intégration de services IntServ

2.2.1 Architecture de qualité de service - IntServ

L'architecture à intégration de services, IntServ décrit dans [15], a été conçue et adoptée à l'IETF pour permettre à internet de supporter la QdS, tout en garantissant un comportement prévisible du réseau vis à vis des applications qui ont des besoins spéciaux en bande passante et/ou en délai. Cette architecture se repose sur le principe de la réservation de ressources.

L'architecture IntServ est subdivisée en un plan de contrôle et un plan de données dont les principales composantes sont :

- Protocole de réservation : Il permet d'effectuer la réservation de ressources pour un nouveau flot qui requiert un niveau de QdS donné. Il met aussi à jour la base de données de contrôle de trafic utilisée par l'ordonnanceur pour déterminer le traitement à offrir aux paquets de chaque flot.
- Contrôle d'admission : Quand un nouveau flot doit être admis dans le réseau, le protocole de réservation invoque le mécanisme d'admission de contrôle. Il détermine s'il y a des ressources suffisantes pour garantir la QdS requise pour le flot.
- Agent de gestion : Pour modifier la base de données de contrôle de trafic qui commande le module d'admission de contrôle pour que ce dernier applique les politiques d'admission de contrôle.
- Protocole de routage : Responsable de la maintenance de la base de données de routage et permet d'indiquer le prochain saut, la prochaine route qui doit être prise pour chaque adresse de destination et pour chaque flot.
- Classification et sélection de route : Pour acheminer les paquets, contrôler le trafic et effectuer la liaison entre les paquets qui arrivent à l'interface du routeur et les classes de trafic.
- L'ordonnancement et mise en file des paquets.

Dans un réseau supportant IntServ, avant qu'un client ne commence à envoyer des paquets dans le réseau, ce dernier envoie une requête vers sa destination, en spécifiant les caractéristiques de son trafic (*traffic specification* ou Tspec). Cette requête est évaluée par le réseau et une réponse est donnée au client pour lui indiquer si oui ou non il peut commencer à transmettre.

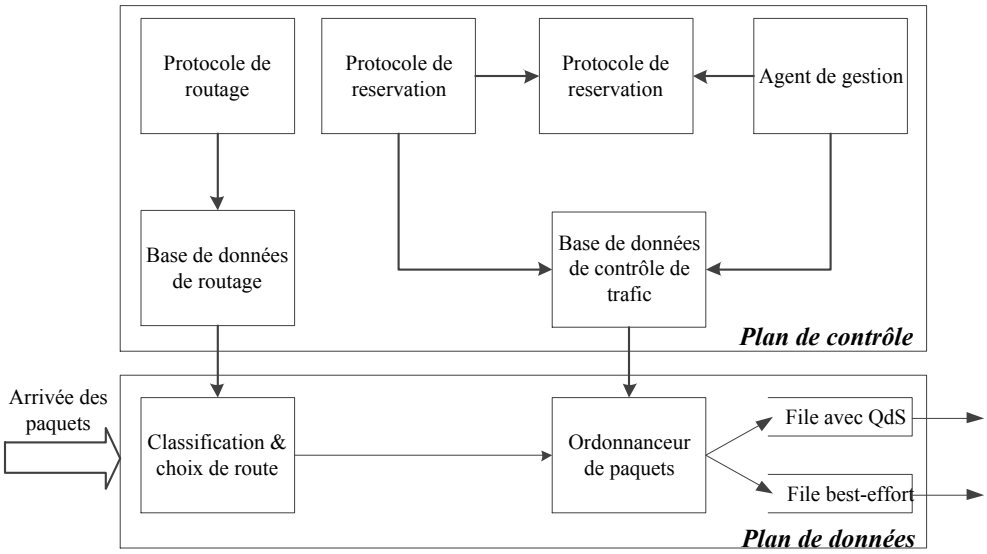


Figure 2.3 Architecture IntServ

Du point de vue du réseau, l'établissement de la communication implique plusieurs étapes qui sont effectuées à travers les principales composantes de l'architecture IntServ montrées à la Figure 2.3.

Dans le plan de contrôle, l'établissement de la communication implique deux tâches principales au niveau du protocole de réservation. D'abord le protocole de routage doit déterminer le prochain hop vers lequel la requête doit être envoyée. Ensuite, se basant sur la décision de routage, l'admission de contrôle doit décider s'il y a assez de ressources disponibles à l'interface de sortie choisie pour accepter la nouvelle communication. Si la réservation est faite, les ressources engagées sont enregistrées dans la base de données de contrôle de trafic. Les informations relatives à la réservation de ressources sont utilisées pour configurer le module de classification et d'ordonnancement dans le plan de données. Quand un paquet arrive à un routeur, le module de classification choisit les paquets qui appartiennent aux flots réservés et les insère dans leurs files d'attente appropriées. L'ordonnanceur sert ensuite les différentes files suivant les informations de la réservation.

Le contrôle d'admission et la réservation de ressources sont basés, entre autres, sur l'information contenue dans le TSpec. La spécification du trafic ou TSpec est une description du trafic pour lequel la requête de service a été effectuée. En général TSpec est une clause du contrat entre les flots de données et le service. Une fois la requête de service est acceptée, le module de service accepte de fournir la QdS demandée tant que les flots de données du trafic demeurent tel qu'ils ont été décrits par le TSpec. Le client désirant faire une réservation doit

effectuer une requête de QdS (*Request Specification* ou RSpec) associée au TSpec,

Deux grandes classes de service sont définies dans IntServ : Service garanti et charge contrôlée. La classe de service garanti est utilisée pour des applications ayant besoin d'une garantie pour le délai et la largeur de bande. Les applications de cette classe de service doivent fournir un TSpec qui indique, entre autres, le débit maximal, la longueur maximale des paquets et la durée des rafales. Le RSpec doit contenir la largeur de bande requise par l'application pour pouvoir déterminer le délai maximal. La classe de service charge contrôlée est utilisée pour des applications n'ayant pas besoin d'une borne garantie mathématiquement pour le délai et la largeur de bande. Pour cette classe de service, on s'assure que l'application recevra un niveau de QdS comparable à celle qu'elle aurait dans un réseau non chargé, mais avec la capacité nécessaire.

Bien que d'autres protocoles puissent être utilisés pour faire les requêtes de QdS dans IntServ, le protocole RSVP [16] qui a été développé pour effectuer la réservation de ressources sur internet, est fortement suggéré [17]. Dans IntServ, RSVP permet à l'application de signaler ses besoins en QdS en transmettant, entre autres, le TSpec, RSpec et le type de service. Deux types de message sont utilisés par RSPV pour établir une réservation : les messages PATH et les messages RESV. Le message PATH va de l'origine à une ou plusieurs destinations et transporte l'information de la classification et le TSpec. Après avoir reçu le message PATH, la destination envoie le message RESV qui contient les informations de la session, le RSpec et la QdS. Une fois le message RESV reçu, l'émetteur peut commencer à envoyer des paquets via le chemin réservé.

Quoiqu'elle puisse permettre de garantir la QdS, l'architecture IntServ présente de nombreux inconvénients qui ont empêché qu'elle soit déployée à grande échelle. D'abord, la granularité est trop petite ce qui peut donner lieu à un très grand nombre de réservation. Ensuite, la quantité d'informations stockées dans chaque nœud augmente avec le nombre de flots, ce qui nécessite beaucoup de capacité de stockage et de traitement au niveau des routeurs. De plus, chaque routeur doit implémenter RSVP, les mécanismes d'admission de contrôle, la classification et l'ordonnancement, ce qui rend le déploiement de IntServ aux routeurs très complexe.

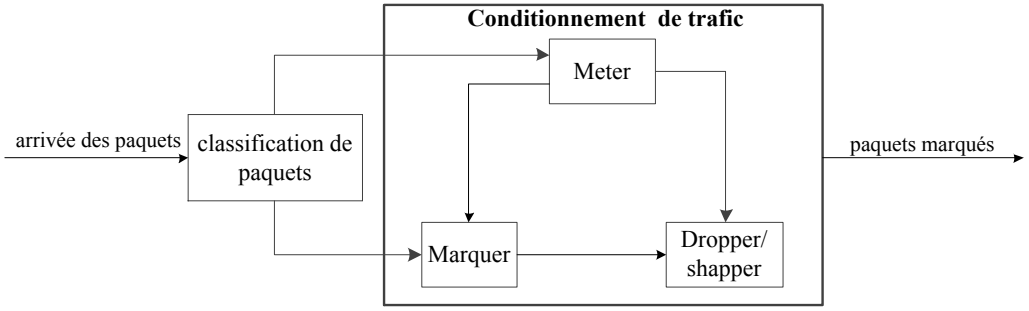
2.2.2 Architecture de qualité de service - DiffServ

Compte tenu des inconvénients de l'architecture IntServ, l'IETF s'est penché sur des mécanismes plus simples pour satisfaire la QdS. Dans cette perspective, l'architecture DiffServ [18] a été proposée.Cette architecture se base sur la plupart des mécanismes élémentaires de QdS décrits dans la section 2.1. Elle définit un cadre qui permet de faire de la QdS dans

les réseaux IP en utilisant des mécanismes qui permettent de classifier les paquets à l'entrée du réseau, les vérifier pour éventuellement effectuer des actions de mises en conformité, les marquer et les transférer à travers les différents nœuds du réseau de façon à garantir le niveau de service spécifié dans le contrat de service (SLA).

L'architecture DiffServ définit des zones de services différenciés. Une zone de service différencié est un domaine constitué de routeurs qui supportent les fonctionnalités de Diffserv. Nous y trouvons deux types de routeurs suivant qu'ils sont situés en bordure de la zone (*Edge Node*) ou à l'intérieur (*Core Node*).

Les routeurs d'entrée (*Edge Node*) sont responsables d'effectuer la classification et le conditionnement du trafic (*Traffic classification and conditionning*).



heightheight

Figure 2.4 Module d'un routeur d'entrée

Classification

La classification des paquets est le mécanisme qui permet d'identifier les paquets et de les regrouper pour constituer des agrégations de flux. Les agrégations de flux constituent les classes de trafic. Généralement cette classification se base sur les informations contenues dans l'entête des paquets tels que le protocole utilisé, les numéros de port et les adresses IP. Elle peut aussi se baser sur des informations statiques telles que la taille des paquets et leur fréquence d'arrivée. Des techniques de DPI (*Deep Packet Inspection*) peuvent être utilisées afin de classer les paquets suivant toutes les informations qu'ils pourraient contenir. Aussi, lorsque les paquets arrivent au routeur d'entrée, ils sont examinés en vu de les regrouper dans des classes de trafic. Tous les paquets appartenant à une même classe se verront attribuer un même code et seront traités de la même façon. Ce code est inscrit dans le champ *Differentiated services* (DS) de l'entête IPv4 ou dans le champ *Traffic class* de IPv6.

Conditionnement de trafic (*Traffic conditionning*)

Le conditionnement de trafic s'assure que, pour un flux donné, le trafic n'excède pas une certaine bande passante et que le profil de trafic pour un flot de trafic donné n'est pas violé. Le module de conditionnement est subdivisé en 4 composantes : *Meter*, *Marker*, *Shaper* et *Policer*, comme l'indique la Figure 2.4.

La composante *meter* joue le rôle de vérificateur de trafic. Il compare le flux de paquets qui entre dans le routeur avec ce qui est spécifié dans le profil du trafic et détermine si un paquet est en conformité avec le profil. Tant que le trafic est conforme au profil qui lui a été attribué, les paquets reçoivent leur priorité et sont transférés. Si le profil est violé, les paquets non-conformes peuvent être soit mis en forme par le *shaper* (lissage de trafic), soit éliminés par le *dropper* (contrôle de trafic) ou marqués. Le marquage effectué lors du conditionnement consiste à attribuer une précédence à un paquet suivant le résultat de la vérification. Si le paquet a été déjà marqué, il s'agit d'un re-marquage.

Plusieurs algorithmes peuvent être utilisés pour faire du conditionnement de trafic, notamment :

- Le seau à jeton (*token bucket*) décrit à la Figure 2.2 ;
- L'algorithme *Single Rate Three Color Marker* (srTCM) qui effectue la vérification d'un flux de paquets IP et qui marque les paquets suivant trois paramètres de trafic : le *Committed Information Rate* (CIR), le *Committed Burst Size* (CBS) et l'*Excess Burst Size* (EBS). Un paquet conforme est marqué vert, jaune s'il est un paquet excédentaire et rouge s'il est non conforme [19] (voir Figure 2.5) ;
- L'algorithme *Two Rate Three Color Marker* (trTCM) effectue la vérification d'un flux de paquets IP et marque les paquets suivant deux paramètres de débit : le *Peak Information Rate* (PIR) et le *Committed Information Rate* (CIR). Un paquet est marqué rouge s'il excède le PIR sinon il est marqué soit jaune, soit vert suivant qu'il excède ou non le CIR [20]. Cet algorithme est représenté à la Figure 2.6.

Les routeurs internes sont constitués d'un module de classification, de plusieurs files d'attente et des modules d'ordonnancement. La classification effectuée par les routeurs internes consiste à regrouper les paquets en fonction de leur étiquette DSCP et à les insérer dans leur file d'attente respective. Chaque file d'attente représente une classe de service. Diffserv définit plusieurs types de classes de trafic qui déterminent le comportement des routeurs (PHB : *Per Hop Behaviour*) vis-à-vis des paquets.

Trois principaux types de comportement ou PHBs ont été définis dans l'architecture DiffServ. Ce sont :

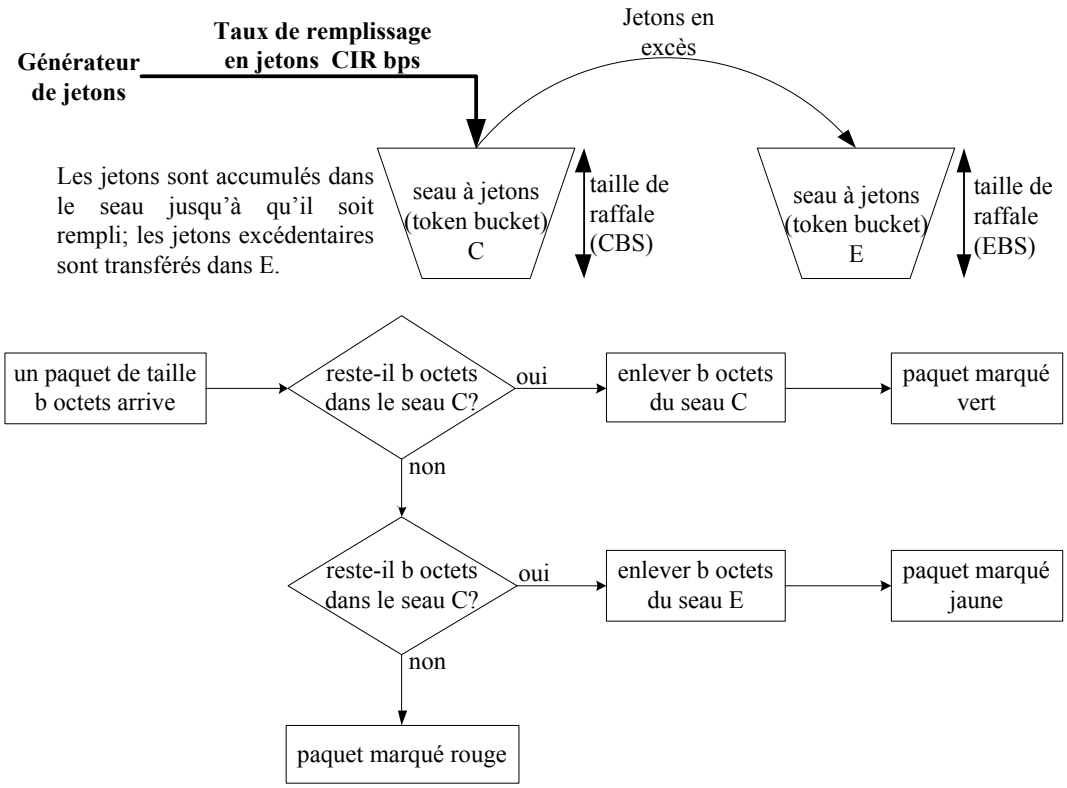


Figure 2.5 Algorithme Single Rate Three Color Marker (srTCM)

1. Le comportement expédié EF [21] qui décrit le comportement d'une classe de trafic où le service est garanti avec un faible débit, un faible taux de perte de paquets et une gigue faible. Il s'assure que le débit du trafic de cette classe soit toujours égal ou supérieur à un certain seuil.
2. Le comportement assuré AF [22] qui est composé de plusieurs classes de services au sein desquelles une autre différentiation qui se base sur la précédence des paquet est introduite. La précédence d'un paquet reflète sa priorité en terme de possibilité pour qu'il soit rejeté par le routeur lorsqu'une situation de congestion se présente. Quatre classes AF ont été définies et pour chacune de ces classes il existe trois (3) niveaux de précédence. Le comportement assuré AF comporte donc 12 PHBs. Aucune relation n'existe entre les classes et il est recommandé que tous les paquets d'un même flux appartiennent à une même classe. Les PHB de la classe AF sont de type AF_{ij} où i identifie la classe et j la précédence.
3. Le comportement DF qui est utilisé pour les trafics de type *best-effort* où la qualité de service n'est pas garantie.

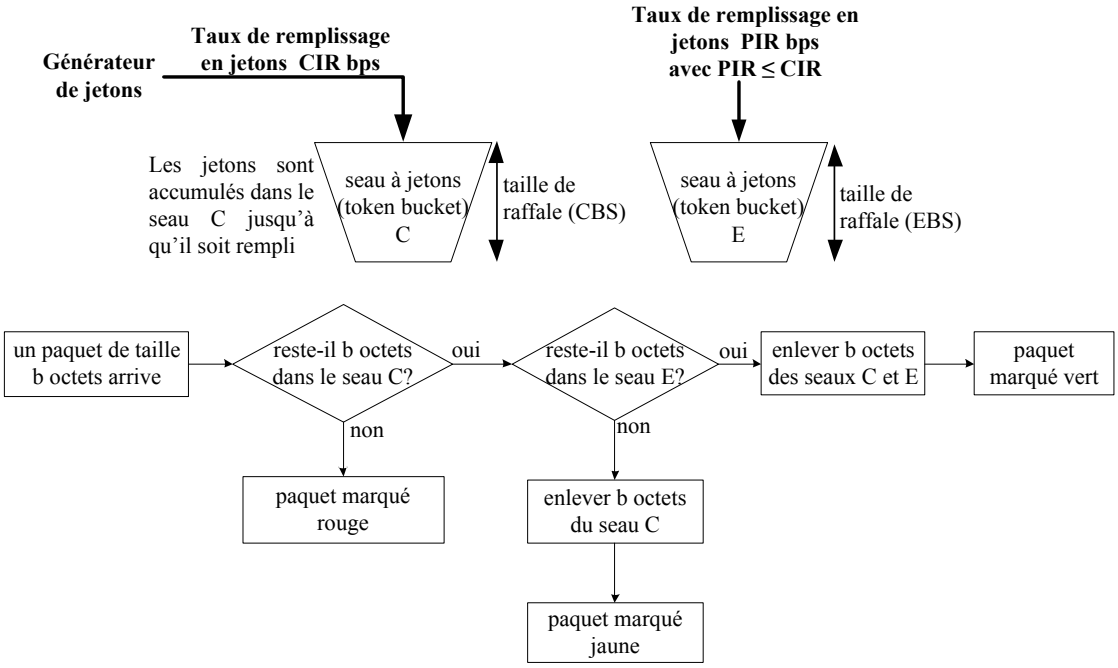


Figure 2.6 algorithme Two Rate Three Color Marker (trTCM)

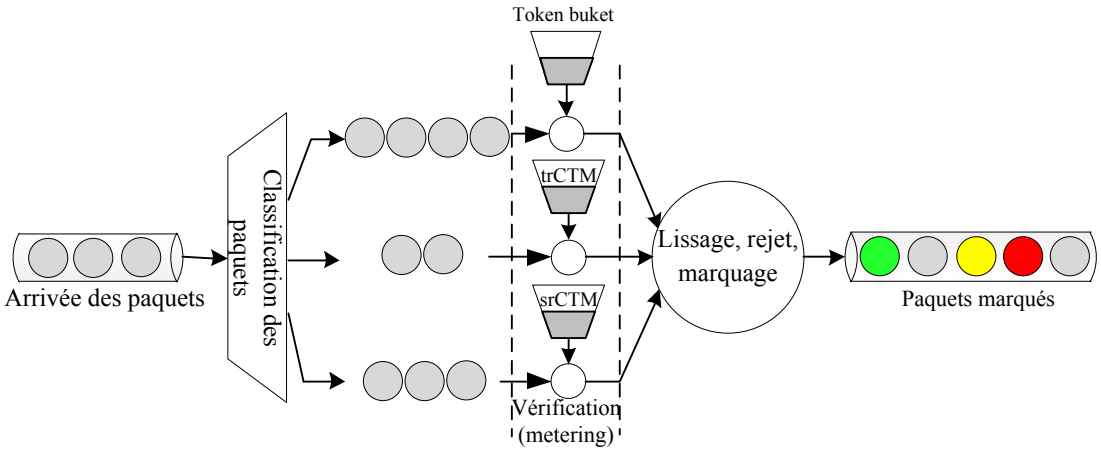


Figure 2.7 Illustration du traitement des paquets à un routeur frontière

Il existe une correspondance entre les files d'attente des routeurs et les différentes classes de trafic. Suivant la classe de trafic, divers algorithmes peuvent être utilisés pour la gestion de la file d'attente correspondante. Pour la classe DF, les algorithmes drop-tail ou RED décrits à la Figure 2.1, peuvent être utilisés. L'algorithme *Weighted Early Drop* (WRED) est recommandé pour les classes AF. Cet algorithme dont le comportement est montré à la Figure 2.8 a été proposé par CISCO pour étendre l'algorithme RED afin de prendre en

compte divers flux de trafic dans une même file d'attente.

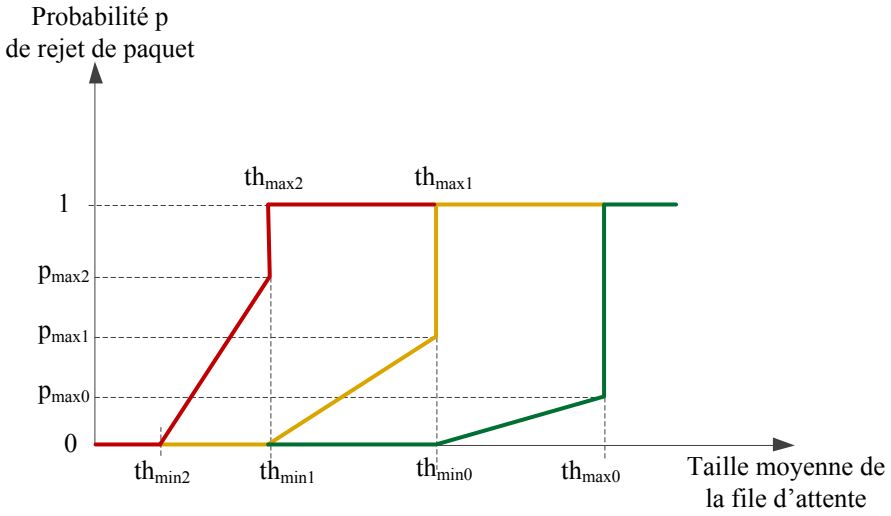


Figure 2.8 Algorithme WRED

L'ordonnancement effectué par les routeurs internes dépend des requis des files d'attente en termes de bande passante ou de retard moyen que doivent expérimenter les paquets. Les algorithmes d'ordonnancement prioritaires, par exemple, peuvent être utilisés pour les flux de trafic EF. Pour les flux AF, les algorithmes d'ordonnancement RR ou WRR et leurs variantes peuvent être utilisés.

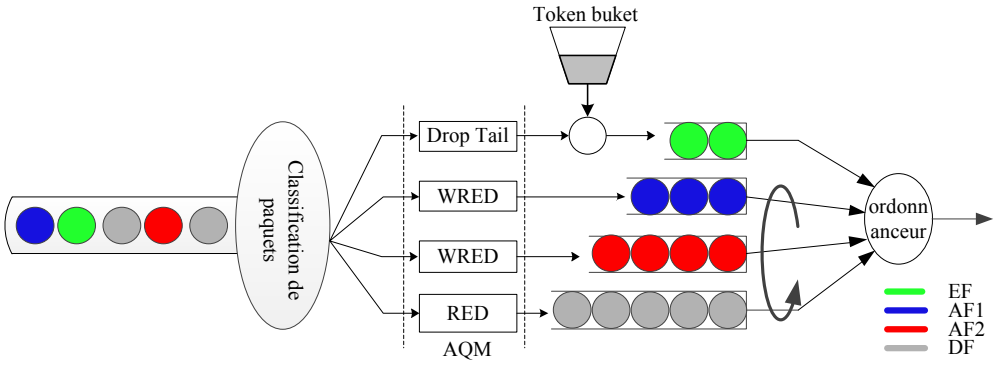


Figure 2.9 Traitement des paquets par un routeur interne

2.3 *Feed-back* et qualité de service

Dans un réseau, le *feed-back* est le moyen par lequel une source peut être informée de l'état du réseau. Actuellement dans les réseaux IP, le *feed-back* est utilisé pour permettre aux routeurs de contrôler la congestion en informant la source d'une situation de congestion naissante dans le réseau. Dans ce cas, la source peut être informée soit par un mécanisme de *feed-back* implicite ou explicite.

Dans le *feed-back* implicite, la source détecte la situation de congestion par des constats qu'elle fait localement comme c'est le cas dans le protocole TCP. Dans TCP, les constats effectués par la source incluent l'expiration du *timer* de retransmission lorsque les acquittements ne sont pas reçus, le retard dans la réception des acquittements et l'arrivée des acquittements dupliqués. Les principales causes des ces situations sont le délai et les pertes de paquets. Dans ce type *feed-back*, les routeurs ne font qu'éliminer des paquets lorsqu'il y a congestion et s'attendent à ce que la source réponde à ces pertes de paquets en réduisant son débit d'émission.

Le *feed-back* implicite qui résulte du rejet des paquets par les routeurs lorsque ceux-ci sont congestionnés présente quelques inconvénients. La perte des paquets constitue un gaspillage des ressources réseau utilisées pour transporter le paquet de sa source au routeur expérimentant la congestion. De plus, une perte de paquet n'implique pas nécessairement qu'il y a congestion.

D'où la nécessite d'avoir un moyen d'indiquer à la source de façon explicite quand une congestion se présente pour qu'elle puisse réduire son débit. Plusieurs propositions ont été faites quant aux mécanismes que les routeurs doivent utiliser pour effectuer la notification. L'un de ces mécanismes est le protocole *Explicit Congestion Notification* (ECN) [23] qui a été adopté à l'IETF.

2.3.1 Notification explicite de la congestion ECN

Le protocole ECN permet à un émetteur d'être informé de manière explicite d'une congestion naissante dans le réseau. Ce protocole se base sur les mécanismes d'*active queue management* où au lieu d'éliminer les paquets lorsque la taille des files d'attente des routeurs atteint un certain seuil, les routeurs les marquent de façon à informer le récepteur que des files d'attente commencent à être remplies. Le récepteur, de son coté, fera l'écho de l'information de la congestion naissante à la source pour que celle-ci anticipe la congestion, en réduisant son débit avant que des paquets ne commencent à être éliminés par les routeurs.

Deux bits de l'entête IP des paquets sont utilisés par ECN pour transporter l'information de la congestion naissante. Le bit *ECN Capable Transport* (ECT) et le bit *Congestion Experienced* (CE). Ce sont les bits 6 et 7 du champ *DSCP* de l'entête IPv4 ou du champ *trafic class* de l'entête IPv6. Ces bits sont codés comme indiqué au Tableau 2.1.

Tableau 2.1 Codage des bits ECN

Bit ECE	Bit CE	
0	0	Pas de ECN
0	1	Source et destination supportent ECN
1	0	Source et destination supportent ECN
1	1	Congestion

Pour identifier la congestion et marquer les paquets, les routeurs utilisent le même algorithme RED décrit au troisième paragraphe de la section 2.1.1 mais modifié pour identifier les différentes combinaisons des bits ECN. Le fonctionnement est le suivant :

- Si la taille moyenne de la file Q_{moy} est inférieure au seuil minimal, th_{min}, le paquet est transmis sans tenir compte du fait que ECN soit activé ou non.
- Si la taille moyenne de la file Q_{moy} est comprise entre th_{min} et th_{max}, le champ ECN de l'entête IP est examiné et suivant la valeur dans le champ ECN, l'un des scénarios suivant se produit :
  - Si $ECN =$ "00", alors la source et la destination ne supportent pas ECN et le paquet est traité suivant l'algorithme RED.
  - Si $ECN =$ "01"ou"10", alors l'algorithme RED détermine si le paquet doit être rejeté. Si l'algorithme indique que le paquet doit être rejeté, alors le routeur met le champ ECN à "11" et le paquet est transmis.
  - Si $ECN =$ "11", alors le paquet est transmis. Cela suppose qu'un routeur en amont commence à expérimenter de la congestion.
- Si la taille moyenne de la file Q_{moy} est supérieure th_{max}, le paquet subit le même traitement que dans RED.

Lorsque la destination reçoit le paquet avec le champ ECN à "11", qui signifie que ce dernier a expérimenté de la congestion, elle en informe la source. Si TCP est le protocole de transport utilisé, l'écho de l'information se fait dans les acquittements,c'est-à-dire paquets ACK envoyés de retour à la source à travers le champ ECE de l'entête du TCP.

2.3.2 *Feed-back* et *User Datagram Protocol* (UDP)

Beaucoup de trafic temps réels utilisent le protocole de transport UDP qui, à la différence de TCP, ne dispose pas de mécanisme d'acquittement. Pour assurer un transport fiable des données temps réel, deux protocoles supplémentaires sont utilisés, RTP [24] et RTCP. Ce sont deux protocoles de la couche application qui s'appuient sur le protocole de transport UDP pour le transport des trafics temps réel. RTP s'occupe de la transmission des données temps réel et RTCP surveille la qualité de service et achemine des informations sur les participants d'une session et sur la qualité de service.

RTCP envoie périodiquement des paquets de contrôle à tous les participants d'une session en utilisant le même mécanisme de distribution que pour les paquets de données dans le but principal de faire l'écho de la qualité de la distribution de données. Pour chaque type d'information transporté par RTCP, un type de paquet différent est utilisé. Ces types de paquet sont :

- SR (Sender Report) qui contient des statistiques de transmission et de réception pour les participants qui sont des émetteurs actifs.
- RR (Receiver Report) qui contient des statistiques de réception pour les participants qui ne sont pas des émetteurs actifs mais récepteurs d'une session.
- SDES (Source Description) qui décrit la source.
- BYE qui permet à une station d'indiquer la fin de sa participation à une session.
- APP qui est un paquet de signalisation spécifique à une application.

Les types de paquet SR et RR sont utilisés pour le *feed-back* sur la transmission des données aux participants d'une session. Ces types de paquets ont le même format et comportent une entête, des informations sur la source qui envoie les paquets SR ou RR, un ou plusieurs blocs de rapport de réception et une extension. Les blocs de rapport fournissent des statistiques sur les données reçues d'une source particulière indiquée dans le bloc de rapport. L'extension est utilisée pour faire l'écho d'autres informations additionnelles spécifiques à une application, qui doivent être rapportées régulièrement sur l'émetteur et le récepteur. Le format d'un paquet SR est illustré à la Figure 2.10.

Un nouveau type de paquet, Extended Report (XR) [25] dont le format est décrit à la Figure 2.11, a été défini pour le protocole RTCP. Les paquets XR offrent de nouvelles possibilités en ce qui concerne le type et la quantité d'informations qui peuvent être transportés. Ces paquets peuvent être utilisés pour de multiples applications. Ils sont constitués d'une entête suivie d'un ou de plusieurs blocs de rapport dont le format est illustré à la Figure 2.12 pour différentes métriques décrivant l'état de la communication (voir Figure 2.11). Ces paquets peuvent être utilisés pour de multiples applications et dans divers contextes. En particulier,

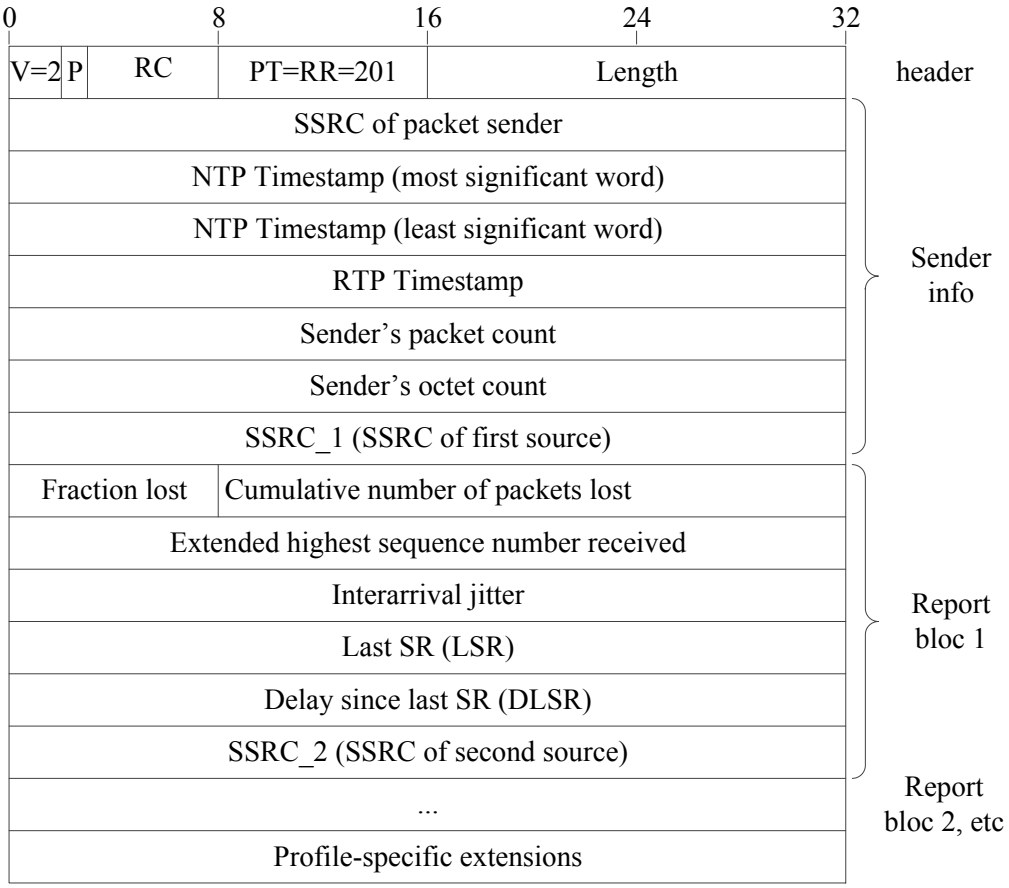


Figure 2.10 Format du type de paquet RTCP SR

il existe un bloc de rapport spécifique pour la voix sur IP illustré à la Figure 2.13.

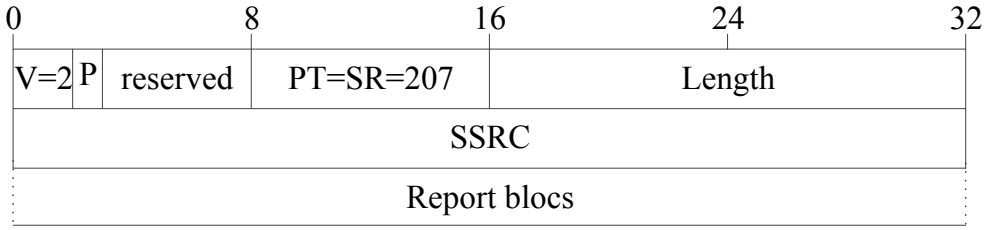


Figure 2.11 Format d'un paquet XR

Dans [26] Westerlund et al ont fait une proposition qui est adoptée à l'IETF sur la façon d'utiliser ECN avec RTP sur UDP en utilisant le protocole RTCP comme mécanisme pour le *feed-back*. Un nouveau bloc de rapport XR est ainsi défini pour contenir les informations

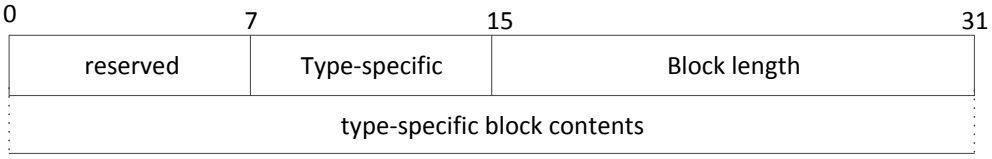


Figure 2.12 Format d'un bloc de rapport XR

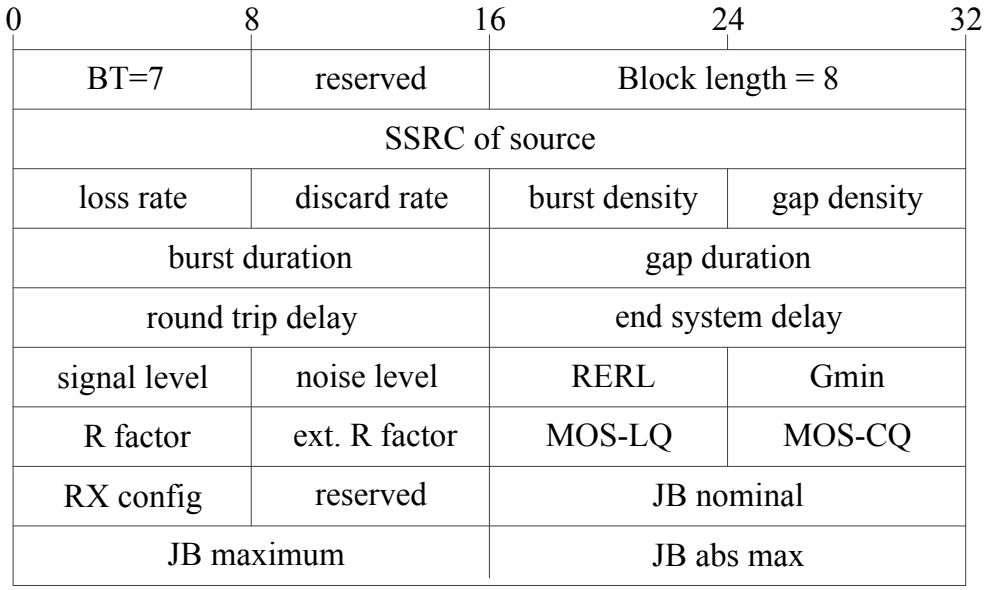


Figure 2.13 Format d'un bloc de rapport VoIP

relatives à ECN.

2.3.3 Réinsértion de la notification explicite de la congestion (Re-ECN)

Dans le protocole ECN, seuls les nœuds expéditeurs et émetteurs peuvent être au courant d'une situation de congestion dans le réseau. Dans certains cas, il peut être utile à un nœud quelconque d'être informé de la congestion en amont ou en aval par rapport à lui même. A l'IETF, le protocole expérimental Re-ECN [27] a été défini dans ConEx [28] qui est un nouveau cadre proposé en vue de gérer la congestion dans internet. Ce protocole, basé sur le protocole ECN, réintroduit dans le réseau le *feed-back* ECN pour permettre à une source d'informer le réseau de la congestion expérimentée par les précédents paquets d'un flux de trafic dans le but :

– de déterminer à partir de n'importe quel endroit du réseau la congestion en amont et en aval de ce point,
– d'imputer la responsabilité de congestion aux sources qui en prennent part,
– de réaliser du contrôle de trafic, etc.

Dans IPv4, Re-ECN utilise le dernier bit non-utilisé de l'entête des paquets IP associé aux deux bits ECN. Pour IPv6, IETF propose d'utiliser une entête d'option de destination [29] puisqu'il ne reste plus de bits disponibles dans l'entête IPv6 des paquets. Le format de cette entête d'option est fourni à la Figure 3.2.

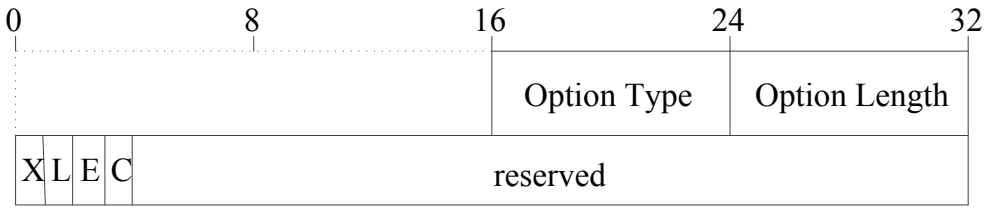


Figure 2.14 Entête de destination utilisée dans ConEx

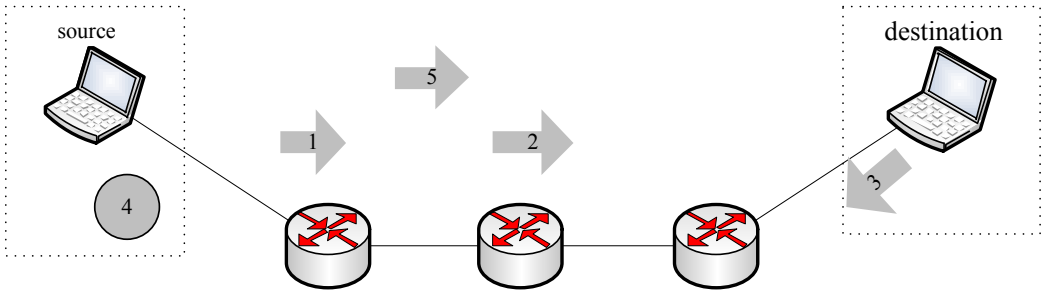


Figure 2.15 Fonctionnement de ConEx

Le fonctionnement du protocole, explicité à la Figure 2.15, est le suivant :

1. La source envoie les paquets appartenant à un flux.
2. Les routeurs sur le chemin marque les paquets au lieu de les éliminer (fonctionnement ECN).
3. La destination renvoie l'information de la congestion expérimentée par les paquets IP en utilisant les segments TCP (notamment les paquets ACK).

4. La source reçoit la notification de la congestion et TCP agit comme si des paquets étaient perdus.
5. La source marque les paquets subséquents en modifiant dans le cas de IPv4 le bit Re-ECN ou la nouvelle entête dans IPv6.

Le réseau peut utiliser cette information dans le cadre de nouveaux mécanismes de contrôle d'admission, de contrôle de trafic, etc.

2.4 Qualité de service et qualité de l'expérience

La QdE prend en compte la perception des utilisateurs, leurs expériences, leurs attentes et la performance des réseaux qui est exprimée typiquement par la QdS. Différents travaux ont été effectués pour évaluer l'impact que certains mécanismes de QdS peuvent avoir sur la QdE et comment ces derniers peuvent contribuer dans certaines conditions à garantir une bonne QdE dans les réseaux IP. La QdS étant déterminée par un ensemble de paramètres objectifs, des propositions ont été faites en vue de trouver une relation entre la QdE et ces paramètres. D'autres propositions se portent sur la mise en œuvre d'architectures permettant de surveiller la QdE pour des applications de type VoIP ou IPTV pour pouvoir réagir lorsque la QdE se dégrade. Une autre catégorie de travaux proposent des mécanismes de QdS dont le fondement est la QdE.

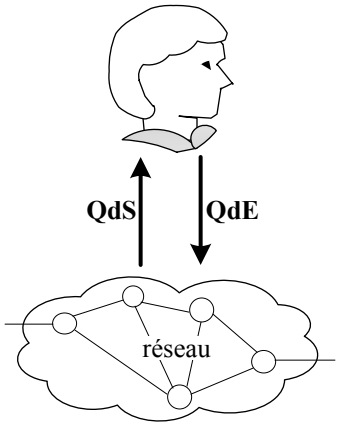


Figure 2.16 QdS et QdE impactent l'une l'autre

2.4.1 Impact des mécanismes de qualité de service sur la qualité de l'expérience

Dans [30], Reguera et al, évaluent l'impact de différents mécanismes de gestion de file d'attente sur la qualité de la voix sur IP. Cependant au lieu de considérer des métriques de réseau (c'est-à-dire perte de paquets, délai et gigue) qui ne sont pas liées à la perception des utilisateurs, comme c'est le cas dans d'autres travaux [31], ils considèrent le MOS qui est la métrique qui exprime la QdE des usagers d'un service donné. Des mécanismes améliorés d'*active queue management* tels que *Adaptative RED* (ARED) [11], *Proportional Integrator* (PI) [32], *Random Exponential Marking* (REM) [33] et *Adaptative Virtual Queue* (AVQ) [34] ont été utilisés. En utilisant l'algorithme *E-model* [7] pour évaluer le MOS, les auteurs ont montré, au moyen de simulations, que l'utilisation des mécanismes d'*active queue management* peut offrir un niveau de satisfaction adéquat aux utilisateurs dans le cas de l'application VoIP. Ainsi la voix sur IP peut tirer parti de ces mécanismes là où il est difficile d'implémenter des techniques traditionnelles de QdS telles que l'utilisation des files de priorité. Particulièrement les stratégies adaptatives comme ARED et AVQ donnent de meilleures performances étant donné qu'ils peuvent régler leurs paramètres suivant les conditions fluctuantes du réseau.

2.4.2 Corrélation entre la qualité de service et la qualité de l'expérience

Il est évident que les problèmes génériques de QdS (par exemple, les pertes, le délai, la gigue, le ré-ordonnancement, les diminutions de débit) impliquent des problèmes génériques de QdE (par exemple, des rayures dans une image vidéo, des artefacts, des temps d'attentes excessifs, etc). Dans cette perspective divers chercheurs se sont évertués à trouver une relation mathématique entre la QdE et la QdS c'est-à-dire trouver une fonction f telle que $QdE = f(QdS)$ avec comme objectif principal le contrôle de la QdE par la QdS.

L'un des premiers travaux [8] réalisé dans ce sens a déduit que la relation entre la QdE et la QdS est non linéaire. Les auteurs ont procédé à des expérimentations dans lesquelles la QdE est considérée comme étant le taux d'annulation CR d'un service internet qui est collecté en tenant compte de différents critères de QdS dans une expérience limitée d'une durée limitée dans le temps. La relation entre CR, représentant la QdE, a été étudiée par rapport à la modification d'un paramètre de QdS dans chaque expérience. Deux paramètres de QdS, la bande passante B et le délai de livraison DT, ont été utilisés. D'après les courbes obtenues, CR qui représente la QdE et la bande passante B qui représente la QdS sont reliés par une fonction logarithmique décroissante de sorte que CR décroît quand B croît :

$$CR = -\alpha_1 \ln B + \beta_1 \tag{2.1}$$

$$QdE = -\alpha_1 \ln QdS + \beta_1 \tag{2.2}$$

Avec α_1, β_1 des paramètres positifs.

D'un autre coté CR et le délai de livraison du réseau DT qui représente la QdS varient linéairement dans le même sens ; c'est-a-dire CR croît linéairement en fonction du délai de livraison :

$$CR = -\alpha_2 DT + \beta_2 \tag{2.3}$$

$$QdE = -\alpha_2 \ln QdS + \beta_2 \tag{2.4}$$

Avec α_2 et β_2 des paramètres positifs.

Cependant aucune relation directe n'a été trouvée entre CR et la latence dans le réseau pour un intervalle donné.

Dans [35], M. Fiedler et al, ont proposé une formule générique selon laquelle la QdE et les paramètres de QdS sont liés par une relation de type exponentiel appelée hypothèse IQX. En général plus la sensibilité subjective de la QdE est élevée, plus élevée est la qualité expérimentée. Si la QdE est très élevée, une petite dégradation entraîne une grande diminution de la QdE, par contre si la QdE est faible, cette même dégradation a un impact moindre. Dans ce contexte, les auteurs supposent que le changement de QdE dépend du niveau actuel de qualité d'expérience, compte tenu du même taux de changement de la valeur de QdS, mais avec un signe différent. Ils aboutissent à la relation suivante :

$$QdE = \alpha \exp -\beta QdS + \gamma \tag{2.5}$$

Avec α ,β et γ des paramètres positifs. La validité de cette relation a été prouvée par des expérimentations.

Une proposition plus récente [5] a établi une corrélation simple entre la QdS et la QdE. Cette corrélation qui se base sur une relation polynomiale d'ordre n telle que :

$$QdE = a_n QdS^n + a_{n-1} QdS^{n-1} + \ldots + a_1 QdS + a_0 \tag{2.6}$$

Ce modèle présente une approximation mais aussi une généralisation des modèles proposés antérieurement. Il vise à faire de la corrélation QdS/QdE le plus simple possible de manière à développer des boucles de contrôle simple pour contrôler la QdE. Cela aura l'avantage de diminuer le coût de calcul au sein du réseau. Il s'adapte également aux différentes formes des courbes de corrélation définies précédemment dans [8] et [5].

2.4.3 Amélioration de la QdE par le *feed-back* de celle-ci

Zizhi et al [36] proposent une méthode pour contrôler la QdS d'une communication de voix sur IP où le codec AMR [9] est utilisé. Le codec AMR est un codeur qui a été développé par l'ETSI et standardisé pour le GSM. C'est un codeur multi-mode dont le débit peut varier entre 4.75 kbps et 12.2 kbps. Cette méthode définit une architecture qui combine la force des techniques de contrôle adaptatif, la technique de QdS de marquage en utilisant une mesure objective de la QdE (c'est-à-dire le MOS qui est calculé objectivement) au lieu de considérer une à une les métriques de QdS. Suivant l'approche utilisée, certains paquets, soit les premiers paquets d'un segment de voix, sont considérés comme prioritaires parce qu'ils sont les plus importants [37]. Pour tenir compte de la priorité de ces paquets, un réseau DiffServ [18] est utilisé et une plus grande priorité leur est attribuée tandis que les autres paquets sont considérés comme faisant partie du trafic best-effort. Par ailleurs, le *feed-back* de la QdE est utilisé pour faire varier le débit du codec AMR à l'expédition.

Moura et al [38] vont dans le même sens que [36] en proposant un algorithme appelé adaMOS qui permet de modifier le débit du codec utilisé à l'émission, en utilisant le *feed-back* du MOS. Il utilise une formule qui donne une approximation du MOS ne dépendant que du délai et des pertes [39]. Cette proposition se diffère de [36] en considérant que tous les usagers n'utilisent pas forcément le même codec.

Dans [40], E. Jammeh et al proposent un nouveau mécanisme d'adaptation basé sur la QdE. À la différence des travaux de Moura [38], ce mécanisme s'applique aux applications de voix et de vidéo sur IP et l'adaptation se fait du côté de l'émetteur et du récepteur. À l'émetteur, l'information de la QdE est utilisée pour soit réduire ou augmenter le débit. Au récepteur, elle est utilisée pour faire varier la taille du tampon de la gigue.

2.4.4 Mécanismes de qualité de service dans les réseaux IP axés sur la qualité de l'expérience

Il a été démontré que la gestion de la QdS en prenant en compte plus de deux critères non corrélés est un problème NP-Complet. Aussi, l'intégration de la QdE dans la gestion de la QdS représente une avenue intéressante en vue de réduire la complexité du problème. D'abord en tant que mesure de performance de bout en bout de la QdS du point de vue de l'utilisateur, la QdE est une mesure importante pour la conception de système et le processus d'ingénierie. Deuxièmement, avec le paradigme de QdE, on peut obtenir de meilleur résultat et ainsi éviter le problème NP-complet car l'objectif est de maintenir la QdE et non optimiser de multiples critères de QdS.

Dans cet ordre d'idée, Hai et al [10] ont développé un mécanisme de routage qui prend en compte la QdE dans le but de développer un mécanisme adaptatif qui peut récupérer l'information de la QdE et l'adapter pour initier des actions conformément au *feed-back* de la QdE. Le système intègre l'évaluation de la QdE dans un système de routage évolutionniste dans le but d'améliorer la perception des utilisateurs basée sur l'algorithme *Q-learning*. Ainsi, il sera possible de choisir les chemins optimaux en terme de QdE dont l'évaluation est faite dans un premier temps par la méthode PSQA [41] et prédite, dans un second temps, par un réseau de neurones. L'approche a été validée en la comparant avec d'autres types d'algorithmes de routage tels que RIP et OSPF, et il fournit de meilleurs résultats pour la QdE des usagers et la quantité de paquet de contrôle est moindre.

Dans [42] un modèle d'optimisation de la qualité de la voix sur IP qui utilise une topologie similaire à un système de contrôle d'admission dans lequel des agents sont gérés par une unité centrale est proposée . Dans ce modèle les agents effectuent un ensemble de mesures dont le délai qui sera utilisé pour évaluer la QdE, en utilisant le *E-model*. Les mesures effectuées sont envoyées à un gestionnaire de la QdS de l'application VoIP. Ce modèle soufre d'un problème d'extensibilité car tous les agents communiquent à un seul nœud.

2.5 Synthèse des travaux

Ces dernières années plusieurs recherches ont été effectuées en vue de mesurer, d'évaluer et d'améliorer la QdE dans les réseaux. Les différents travaux effectués proposent des mécanismes qui, pour la plupart, ne réagissent pas lorsque la QdE de l'usager se dégrade. Ceux qui le font ne mettent l'accent que sur les applications de type voix sur IP ou vidéo. en proposant surtout des mécanismes pour faire varier le débit des codeurs lorsque la QdE se dégrade, mais ces mécanismes ne peuvent pas s'appliquer aux autres types d'applications. Pour les mécanismes de QdS axés sur la QdE, il en existe très peu et ils sont le plus souvent très difficiles à mettre en œuvre.

Dans le chapitre suivant, nous allons dans un premier temps proposer une nouvelle méthode d'avoir le *feed-back* de l'information de la QdE et de l'introduire dans le réseau de sorte que, le cas échéant, les routeurs puissent l'utiliser pour ajuster les mécanismes de QdS. Dans un second temps, un nouveau mécanisme de traitement des paquets par les routeurs à l'intérieur d'un domaine DiffServ. Ce nouveau mécanisme tiendra compte de l'information de la QdE disponible.

CHAPITRE 3

MÉCANISME DE QUALITÉ DE SERVICE AXÉ SUR LA QUALITÉ DE L'EXPÉRIENCE

Bien que le fait de satisfaire tous les paramètres recommandés de qualité de service(QdS) d'une application ne puisse garantir que les usagers seront satisfaits, il n'en demeure pas moins vrai que la qualité de service a un impact sur la QdE, comme certains des travaux présentés dans le chapitre 2 l'ont montré. Ainsi, pour répondre à l'attente des usagers, l'implémentation de la QdS dans les réseaux actuels doit être axée sur la QdE et doit fournir les niveaux de performance nécessaire pour avoir une bonne QdE.

Dans les prochaines sections de ce chapitre nous allons décrire en détail les mécanismes que nous proposons pour faire le *feed-back* de la QdE et l'acheminer dans le réseau pour être utilisée dans l'implémentation de mécanisme amélioré de QdS. Le mécanisme de QdS proposé y sera également développé.

3.1 Mécanisme de *feed-back* de la QdE

Comme décrit à la section 2.3 du chapitre précédent, les mécanismes de *feed-back* sont utilisés dans les réseaux IP, notamment le protocole ECN, pour notifier une source de la congestion rencontrée par ses paquets pour qu'elle puisse adapter son débit de transmission et autres paramètres d'encodage. Mieux encore, cette notification peut être ré-insérée dans le réseau pour permettre aux routeurs le long du chemin de déterminer la congestion en amont et en aval et ainsi être capable de faire du contrôle de trafic. De même, il est possible dans le cas d'une communication donnée qu'un récepteur puisse informer un émetteur du niveau de la QdE pour un flot trafic donné ; ce qui permettra à l'émetteur d'ajuster les paramètres de transmission sur lesquels il peut agir et même ré-insérer l'information de la QdE, de sorte que les routeurs entre l'émetteur et le récepteur en tiennent compte dans différents mécanismes de QdS.

Étant donné une communication impliquant un récepteur et un émetteur, la technique proposée pour le *feed-back* de la QdE se base sur l'hypothèse selon laquelle, au niveau du récepteur, la QdE peut être évaluée à un intervalle régulier pour un trafic donné.

Les requis sont :

1. Le mécanisme doit être utilisable pour tout type d'applications pour lequel la QdE peut être déterminée. Ces applications incluent notamment le web, la voix sur IP (VoIP), la vidéo et les jeux vidéo en ligne.
2. Il ne doit pas dépendre de la méthode utilisée pour évaluer la QdE.
3. Il doit se faire '*in band*'. Autrement dit, il est souhaitable que l'information de la QdE soit disponible dans l'entête des paquets de données pour assurer l'extensibilité tout en évitant de générer de la signalisation supplémentaire.
4. Il doit utiliser les protocoles existants autant que possible pour que son implémentation puisse se faire sans modifier l'architecture de l'Internet.

L'Internet se repose aujourd'hui sur l'architecture protocolaire TCP/IP dans laquelle deux protocoles de transport sont utilisés. Le protocole TCP, orienté connexion, qui est utilisé pour les application requérant une remise séquentielle et fiable, et le protocole UDP qui est utilisé surtout dans le cas des applications multimédias. Ainsi le mécanisme proposé pour faire le *feed-back* de la QdE tient compte de ces deux protocoles de transport.

3.1.1 *Feed-back* de la QdE avec TCP

Dans le cas d'une application qui utilise TCP comme protocole de transport, le renvoi de la QdE à l'émission se fait en faisant du 'piggyback' qui consiste à utiliser le même segment pour envoyer des données et acquitter les données reçues. En effet, le protocole TCP permet de transférer des données dans les deux directions d'une connexion en utilisant les paquets *ACK* pour transporter des données. Ainsi, pour les trafics TCP, la QdE sera calculée au récepteur ou à un nœud distinct du récepteur et sera renvoyée à l'émetteur via les ACK dont la charge utile contiendra les informations relatives à la QdE.

3.1.2 *Feed-back* de la QdE avec UDP

Pour certaines applications, UDP est le protocole de transport qui est utilisé. C'est le cas, par exemple, des applications multimédias. Comme décrit dans le chapitre précédent à la section 2.3.2, deux autres protocoles sont utilisés par dessus de la couche transport pour ces types d'application. Ce sont les protocoles RTP et RTCP (voir section 2.3.2). RTP est utilisé pour le transport des données et RTCP pour le *feed-back* des informations de la qualité de la communication entre les intervenants. Plusieurs types de paquets ont été définis pour RTCP en particulier le type de paquet XR qui permet de définir divers blocs de rapport pour transporter des informations de *feed-back* à différents participants d'une session (voir section

2.3.2. Par exemple, Westerlund et al [26] ont défini un nouveau bloc de rapport XR pour effectuer le *feed-back* de ECN pour RTP par dessus de UDP.

De même, pour faire l'écho vers l'émetteur de l'information de la QdE, nous proposons d'ajouter un nouveau bloc de rapport au paquet XR dédié au transport de l'information de la QdE. Ce bloc de rapport comprendra entre autres l'information de la QdE exprimée par le MOS ou une autre métrique de QdE, le sens et le taux de variation. Le format de ce bloc de rapport est présenté à la Figure 3.1.

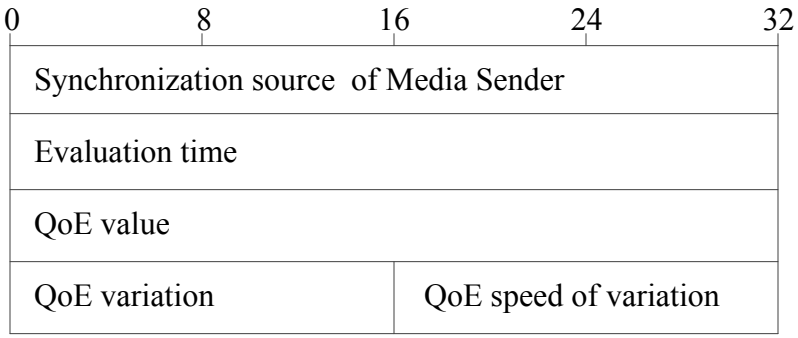


Figure 3.1 Exemple de format d'un bloc de rapport XE pour le *feed-back* de la QdE

3.1.3 Ré-insertion de la QdE dans le réseau

L'information de la QdE, une fois calculée au récepteur et renvoyée à l'émetteur, doit être introduite dans le réseau pour qu'elle parvienne aux différents nœuds du réseau pour être utilisée, le cas échéant, dans les politiques de QdS. Dans cette proposition, cela se fait à travers l'entête des paquets IP et le champ designé à ConEx/Re-ECN.

Dans l'entête IPv4, puisqu'il n' y a qu'un seul bit disponible (voir section 2.3.3), l'information complète de la QdE ne peut être transportée dans l'entête du paquet. Le dernier bit disponible sera utilisé pour signaler aux routeurs que la QdE s'est dégradée de sorte qu'ils puissent déclencher le mécanisme prévu à cet effet. Ce bit sera à 1 s'il y a dégradation pour indiquer aux nœuds intermédiaires que la QdE du flot auquel le paquet appartient s'est dégradé et qu'ils doivent adapter les mécanismes de QdS en conséquence. Sinon, le bit est à 0. Toutefois, cela nécessite que l'émetteur puisse, à partir des informations relatives à la QdE qui lui a été envoyées par l'expéditeur, déterminer s'il faut agir sur les mécanismes de QdS dans le réseau. Dans ce cas, il ne peut y avoir de cohabitation entre le mécanisme proposé et le protocole Re-ECN décrit à section 2.3.3.

Avec le protocole IPv6, deux options sont proposées. La première consiste à utiliser les bits restants de l'entête option de destination de IPv6 proposé pour le nouveau protocole Re-ECN défini dans le cadre de ConEx (voir section section 2.3.3 du chapitre précédent). Cette entête fait 32 bits de longueur et ConEx en utilise 4 pour Re-ECN et les autres sont réservés. Une partie des 28 bits réservés sera utilisée pour acheminer l'information de la QdE aux nœuds intermédiaires entre la source et la destination. Le format de cette entête de destination est fourni à la Figure 3.2.

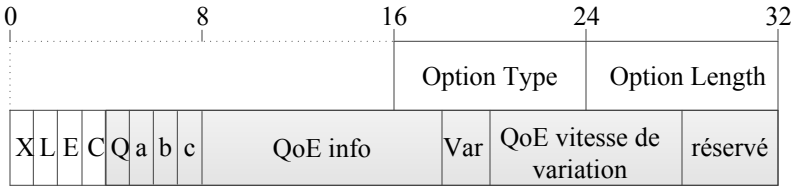


Figure 3.2 Entête de destination utilisée dans ConEx

Les bits X, L, E, C sont utilisés par ConEx. Le bit Q permet d'indiquer si oui ou non l'information de la QdE doit être prise en compte au niveau des routeurs. Les bits a, b, et c, quand ils sont à 1, indiquent que l'information de la QdE, le sens de variation, ainsi que la vitesse avec laquelle l'information varie sont présents. La valeur de la QdE est codée avec les dix prochains bits. Les deux bits suivants permettent de coder le sens de variation et les 8 prochains bits sont utilisés pour le taux de variation.

Si Re-ECN n'est pas utilisé, une nouvelle entête de destination sera ajoutée à l'entête IPv6. Cette entête aura le même format que celle utilisée dans ConEx sauf qu'elle sera privée des bits relatifs à ConEx. Mentionnons que plus de bits peuvent être utilisés dans l'entête de destination car elle n'est pas limitée à 32 bits.

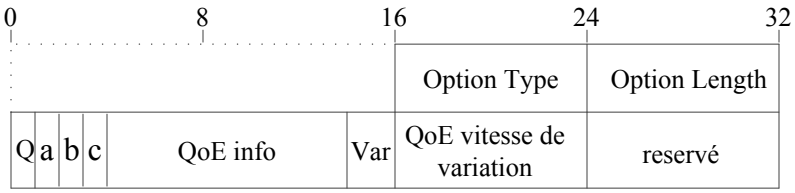


Figure 3.3 Entête de destination pour le *feed-back* de la QdE sans ConEx

3.1.4 Format de l'information du *feed-back* de la QdE

L'information de la QdE peut être retournée sous divers formats comme présenté au Tableau 3.1 :

Valeur de QdE

Comporte la valeur actuelle de la métrique de QdE (MOS, facteur R). Toutefois, il est recommandé de normaliser toutes les autres métriques de QdE en la métrique du MOS variant entre 1 et 5. Dans cette proposition, la valeur du MOS, est codée sur dix (10) bits. Trois (3) des 10 bits sont utilisés pour coder la partie entière et les sept (7) autres pour coder la partie décimale.

Variation de la QdE

Indique le sens de la variation de l'information de la QdE entre au moins deux (2) intervalles d'évaluation de la QdE et permet de savoir, en moyenne, si la QdE a dégradé ou amélioré. Cette information donne un aperçu sur la fluctuation de la QdE.

Vitesse de variation de la QdE

Normalisée pour avoir ses valeurs comprises en 0 et 100 et exprimée en pourcentage, ce champ indique le rythme moyen de variation de l'information de la QdE.

Bit de signalisation de la QdE

Il est possible d'utiliser un seul bit pour indiquer qu'il y a dégradation de la QdE et le besoin d'améliorer ou d'ajuster les mécanismes de QdS en vue d'améliorer la QdE (comme avec IPv4).

Tableau 3.1 Format de l'information de la QdE

QdE info	Description	Exemple de valeurs
Valeur de la QdE	L'actuelle valeur de la métrique QdE(exemple MOS)	3.60(0110111100)
Variation de la QdE	Indique si la QdE décroît ou non	11
Vitesse de variation de la QdE	Rythme auquel l'information change	40%(00101000)
Bit de signalisation	Bit utilisé pour indiquer la nécessité d'améliorer le QdE (uniquement pour IPv4)	1

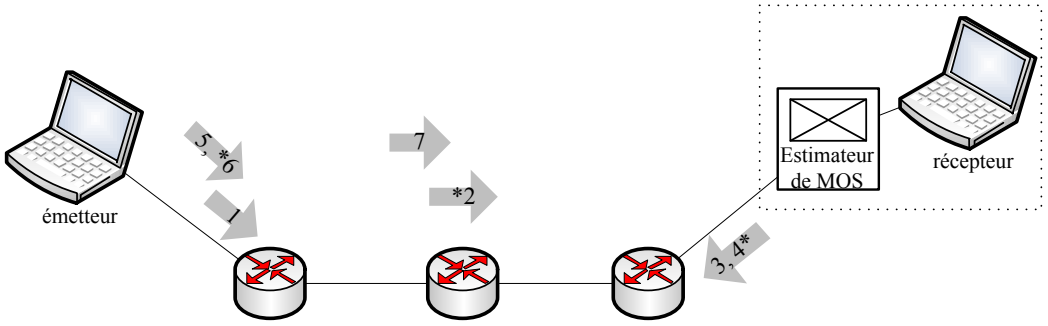


Figure 3.4 Mécanisme de *feed-back* proposé

La Figure 3.4 décrit le fonctionnement global du mécanisme proposé utilisé conjointement avec ConEx. Les étapes marquées du signe '*' sont des procédures de ECN/ConEx et sont optionnelles.

1. La source envoie des paquets qui appartiennent à un flux de trafic.

*2. Les routeurs entre la source et la destination marquent les paquets lors d'une situation de congestion dans le cas de ECN.

3. Le récepteur évalue et renvoie les informations relatives à la QdE de l'utilisateur à la source, pour le flux en question. La QdE peut aussi être évaluée par un autre nœud autre que le récepteur.

*4. Si ECN est utilisé, le récepteur notifiera également l'émetteur lorsque des paquets ont rencontré de la congestion.

5. La source fait parvenir dans l'entête IP l'information de la QdE qui a été renvoyée par le récepteur aux routeurs pour que ces derniers puissent l'utiliser lors du traitement des paquets.

*6. La source marque l'entête ConEx des paquets subséquents, avec l'information de ECN (valable uniquement avec IPv6).

7. Les routeurs utilisent l'information de la QdE associée aux paquets pour mieux ajuster les divers mécanismes de QdS. Par exemple, les routeurs frontière peuvent l'utiliser dans des mécanismes améliorés de contrôle d'admission.

3.2 Mécanisme de QdS proposé basé sur la QdE

La principale motivation pour un réseau IP de suivre la QdE des usagers et d'en avoir le *feed-back* est de pouvoir réagir lorsqu'elle se dégrade ou de prévenir qu'il y ait dégradations.

De ce fait, il faut développer des mécanismes de QdS qui puissent prendre en compte la QdE. C'est dans ce sens qu'est proposé ce nouveau mécanisme de QdS basé sur la QdE dans le cadre de l'architecture DiffServ.

Certaines recommandations ont été faites sur la manière d'utiliser DiffServ en vue de satisfaire la QdS pour certaines applications. Il est recommandé, par exemple, pour les applications de voix sur IP d'utiliser la classe EF [21] dont la file est gérée suivant l'algorithme d'ordonnancement avec priorité *priority queuing* (PQ). Cet algorithme est utilisé de façon à garantir une faible latence, un faible taux de perte de paquets et une faible gigue. Or, l'algorithme PQ, permettant d'accorder une priorité absolue à une file au détriment des autres, peut provoquer de la famine pour les autres files. Pour éviter le risque de famine, un limiteur de bande passante (*rate limiter*) est appliqué à la file EF. Cependant, cette approche présente de nombreuses limitations. Les trafics catégorisés dans EF peuvent avoir des caractéristiques diverses et des requis de QdS différents qui varient dans le temps. Cela réduit l'efficacité du réseau puisque les conditions les plus rigoureuses de QdS doivent être respectées pour tous les trafics EF. Par ailleurs, le fait de limiter le trafic dans EF réduit la flexibilité de l'allocation. En conséquence, tout le trafic EF qui arrive après que la file EF ait atteint sa limite est éliminé même s'il y aurait de la capacité résiduelle dans les autres files. D'où une solution plus flexible s'impose.

3.2.1 Vue d'ensemble du mécanisme proposé

Le mécanisme proposé s'inscrit dans le cadre de l'architecture de QdS DiffServ et consiste à prendre en compte le *feed-back* de la QdE dans la classification qui est effectuée par les routeurs internes à un domaine DiffServ.

Comme décrit à la section 2.2.2 du chapitre précédent, lorsque les paquets préalablement marqués par les routeurs situés à la frontière d'un domaine DiffServ arrivent à un routeur interne, le routeur lit le champ DSCP de l'entête du paquet et insère ce dernier dans la file d'attente correspondante. Dans le mécanisme proposé, un traitement particulier est fourni à certains trafics qui sont considérés comme sensibles à la QdE. Un trafic sensible à la QdE est un trafic pour lequel un seuil minimum de QdE doit être garanti. Ces trafics peuvent provenir de n'importe quelle application où la QdE peut être évaluée, telle que la voix sur IP, l'IPTV, les jeux vidéos en ligne ou même le web. La proposition se base sur le mécanisme décrit à la section 3.1 qui permet aux routeurs d'avoir accès aux informations de la QdE. Ainsi, la classification de ces paquets dépendra non seulement du code DSCP mais aussi de l'information de la QdE présente dans l'entête réseau des paquets.

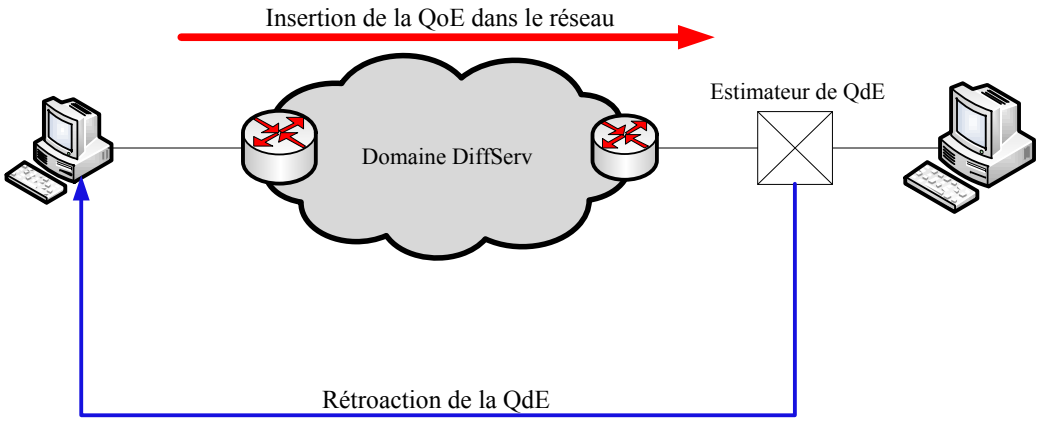


Figure 3.5 Vue d'ensemble

3.2.2 Mécanisme de traitement des paquets

Le mécanisme de traitement des paquets décrit la manière dont les paquets sont traités lorsqu'ils arrivent aux routeurs cœur du réseau DiffServ et suppose qu'ils ont été traités de façon appropriée par les routeurs frontière et qu'un code DSCP leur a déjà été attribué. Ce mécanisme modifie la méthode de classification des paquets effectuée par les routeurs cœur qui consiste, en fait, à aiguiller les paquets vers leurs files d'attentes respectives. Dans DiffServ, vue la correspondance qui existe entre les classes de trafic et les files d'attente, tous les paquets d'une même classe sont toujours insérés vers la même file d'attente. Il s'agit là d'un aiguillage statique comme illustré à la Figure 3.6.

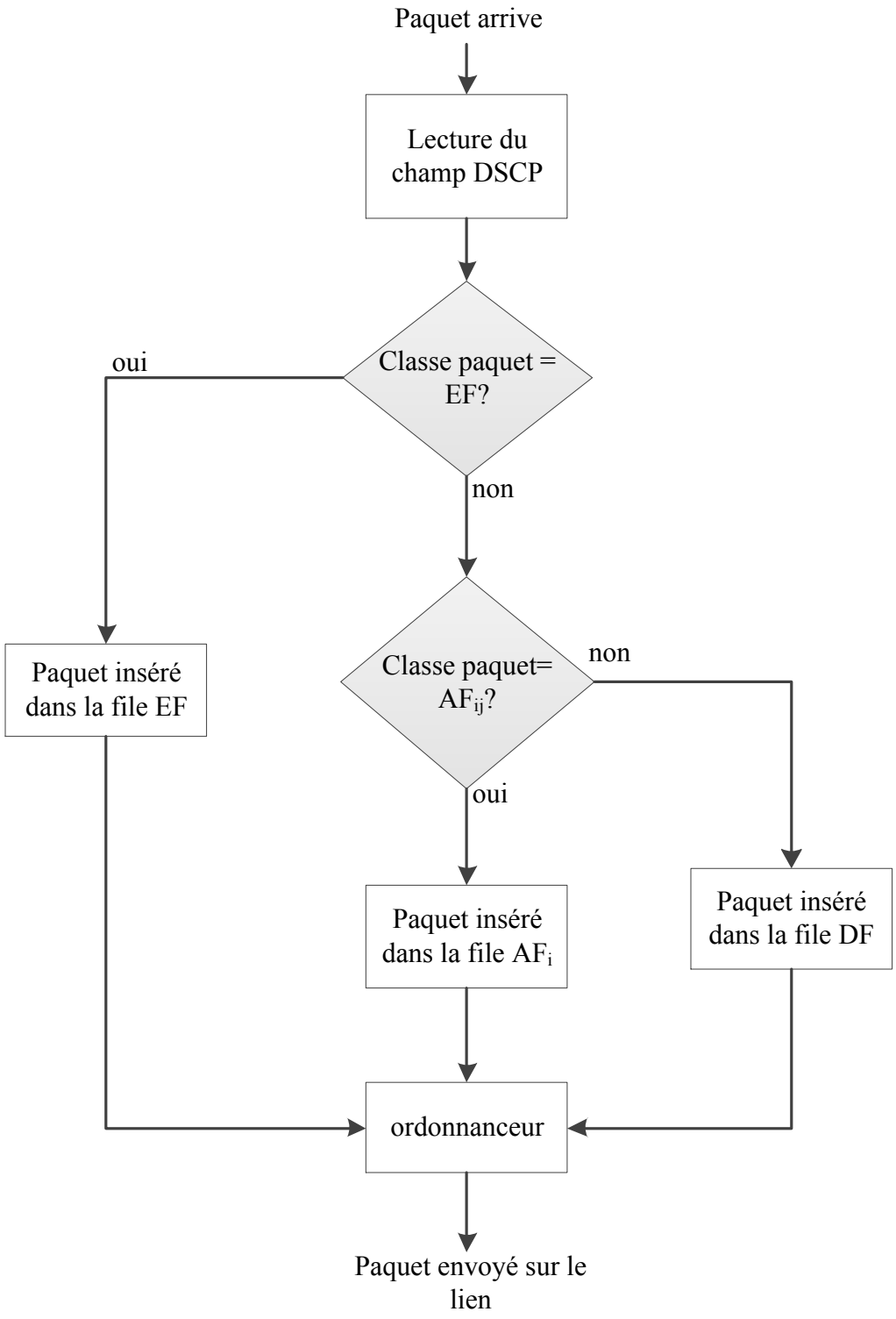


Figure 3.6 Classification des paquets dans un routeur interne à un domaine DiffServ

Dans le nouveau mécanisme, l'aiguillage se fait de manière dynamique et dépend, outre du code DSCP du paquet, de l'information de la QdE et s'applique spécifiquement aux classes AF soit, AF1, AF2, AF3 et AF4. Un ordre de priorité est établi entre les différentes files AFi avec $i \in \{1, 2, 3, 4\}$ dans l'allocation de la bande passante. En effet, soit wi le poids associé à la file AFi, ce poids indique la proportion de bande passante allouée aux trafics de la classe AFi. Il faut que :

$$w1 < w2 < w3 < w4 \tag{3.1}$$

Comme décrit dans la Figure 3.7, lorsqu'un paquet arrive à un routeur, s'il est de classe EF ou DF, il est transféré vers la file EF ou DF comme dans DiffServ classique. S'il est de classe AF, soit AFi avec $i \in \{1, 2, 3, 4\}$, et que le niveau de la QdE ne descend pas au-dessous d'un certain seuil, le paquet est inséré dans la file correspondant à sa classe. Sinon, si le paquet est de classe AFi avec $1 < i \leq 4$, l'état des files AFj avec $j > i$ est vérifié et est comparé à l'état de la file d'attente correspondant à la classe du paquet. L'état d'une file d'attente à un instant donné correspond à la taille de la file d'attente et le délai qu'expérimentera le prochain paquet qui y sera inséré. S'il existe une file AFj avec $i \neq j$ qui permet au paquet d'expérimenter un meilleur temps d'attente et qui réduit sa probabilité d'être éliminé, le paquet y sera inséré sans que la valeur du champ DSCP de l'entête du paquet ne soit modifiée. Pour ne pas détériorer la qualité de service des flots de classe AFj, des seuils peuvent être fixés pour les différents paramètres de la file AFj. Parmi les paramètres qu'il faut fixer, il y a le niveau de remplissage d'une file à un instant donné pour lequel un seuil peut être fixé de sorte que même si la file j serait la meilleure file vers laquelle le paquet doit être aiguillé, si le niveau de remplissage dépasse ce seuil, le paquet n'y sera pas aiguillé.

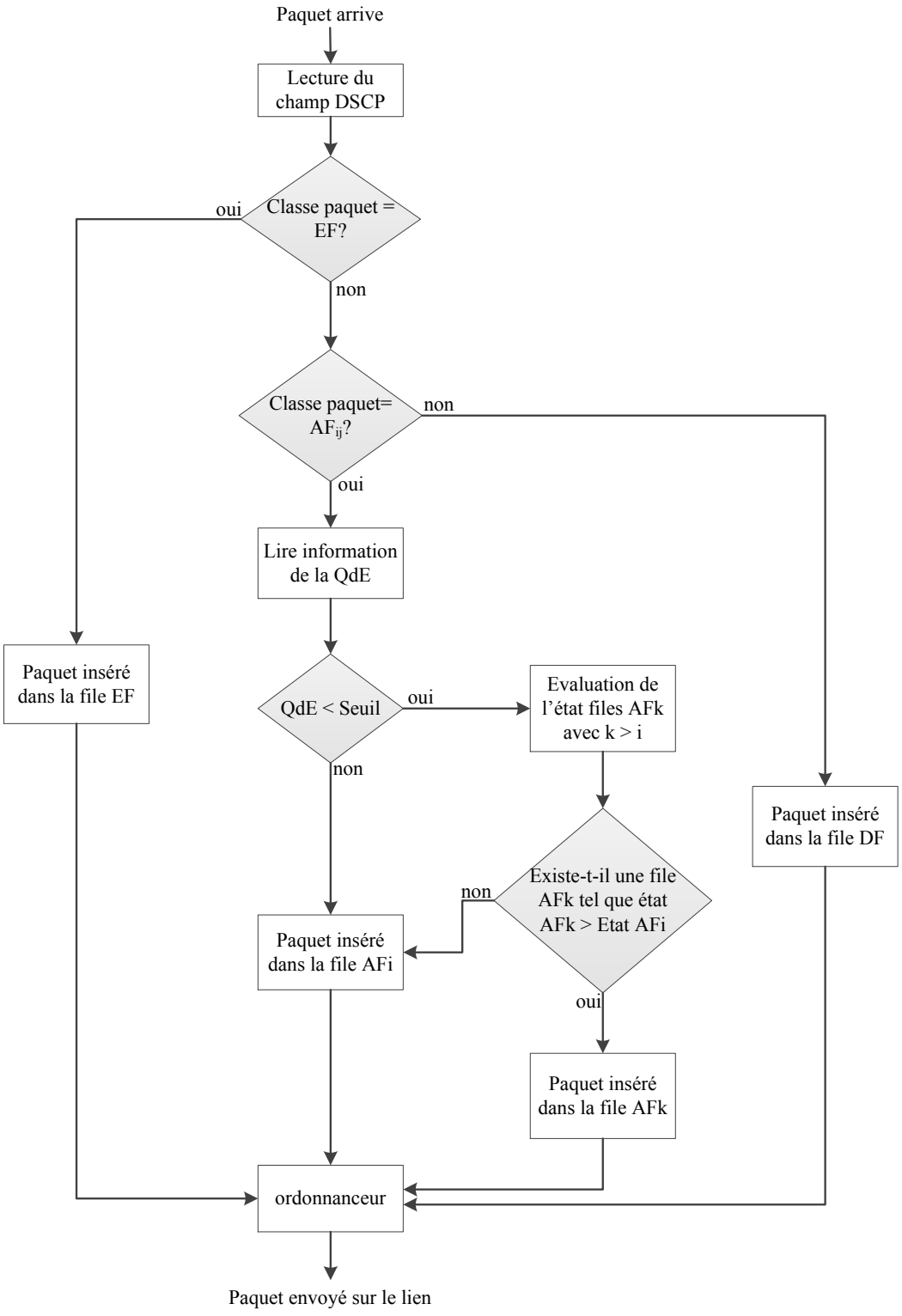


Figure 3.7 Classification des paquets dans le mécanisme proposé

Dans le cadre de cette proposition, la Figure 3.8 donne une une illustration du cheminement des paquets à travers les files d'attente d'un routeur interne à un domaine DiffServ.

Les paquets rouges sont de classe AF2, cependant le paquet 9 est inséré dans la file AF1 puisque les informations de la QdE du flot auquel il appartient et l'état de la file AF1 laissent prévoir que le paquet ne sera pas éliminé et qu'il expérimentera un délai moindre.

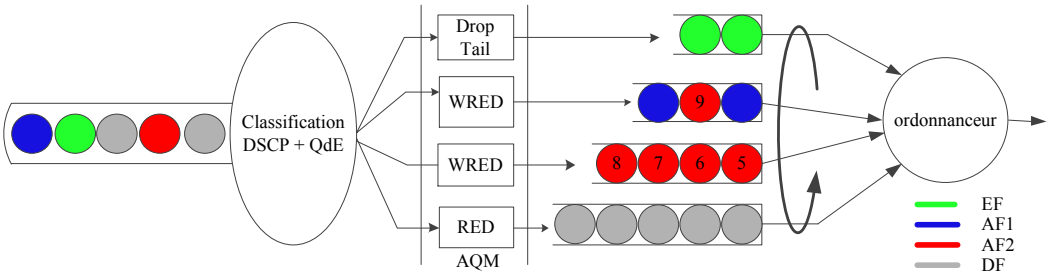


Figure 3.8 Illustration de la classification des paquets dans le mécanisme proposé

La Figure 3.9 présente un diagramme simplifié de traitement quand un paquet arrive à l'interface d'un routeur jusqu'à ce qu'il arrive à l'interface de sortie. En supposant que le réseau utilise IPv6 à la couche réseau, le routeur lit le bit Q de l'entête d'option de destination de IPv6 définie dans ConEx ou dans celle définie dans cette proposition (Étape 1 de la Figure 3.9). Si le bit Q est à 1, alors le mécanisme continue (Étape 2a). Sinon le paquet est traité suivant la classification de DiffServ (Étape 2b) et le paquet est inséré dans le file correspondant à sa classe jusqu'à ce qu'il atteigne l'ordonnanceur (Étape 4). Pour, les paquets des classe EF et DF, Q sera toujours à 0 puisque le mécanisme ne s'applique pas aux paquets de ces classes. Le mécanisme se poursuit avec le routeur qui examine le code DSCP du paquet, les informations relatives à la QdE et l'état des files d'attente (Étape 3). Si l'évaluation de la QdE et l'état des files d'attente indiquent que le paquet peut être inséré dans une meilleure file d'attente (Étape 3a), le paquet est inséré dans celle-ci pour être traité ensuite par l'ordonnanceur (Étape 4). Cela se fait sans attribuer un autre code DSCP au paquet (Étape 3b). Sinon, le paquet est inséré dans la file correspondant à sa classe. L'évaluation de la QdE se fait en considérant les informations de QdE disponibles. Au plus, trois (3) informations de QdE peuvent être disponibles. Ce sont la dernière valeur, la variation et la vitesse de variation de la QdE pour le flot auquel appartient le paquet(voir Tableau 3.1).

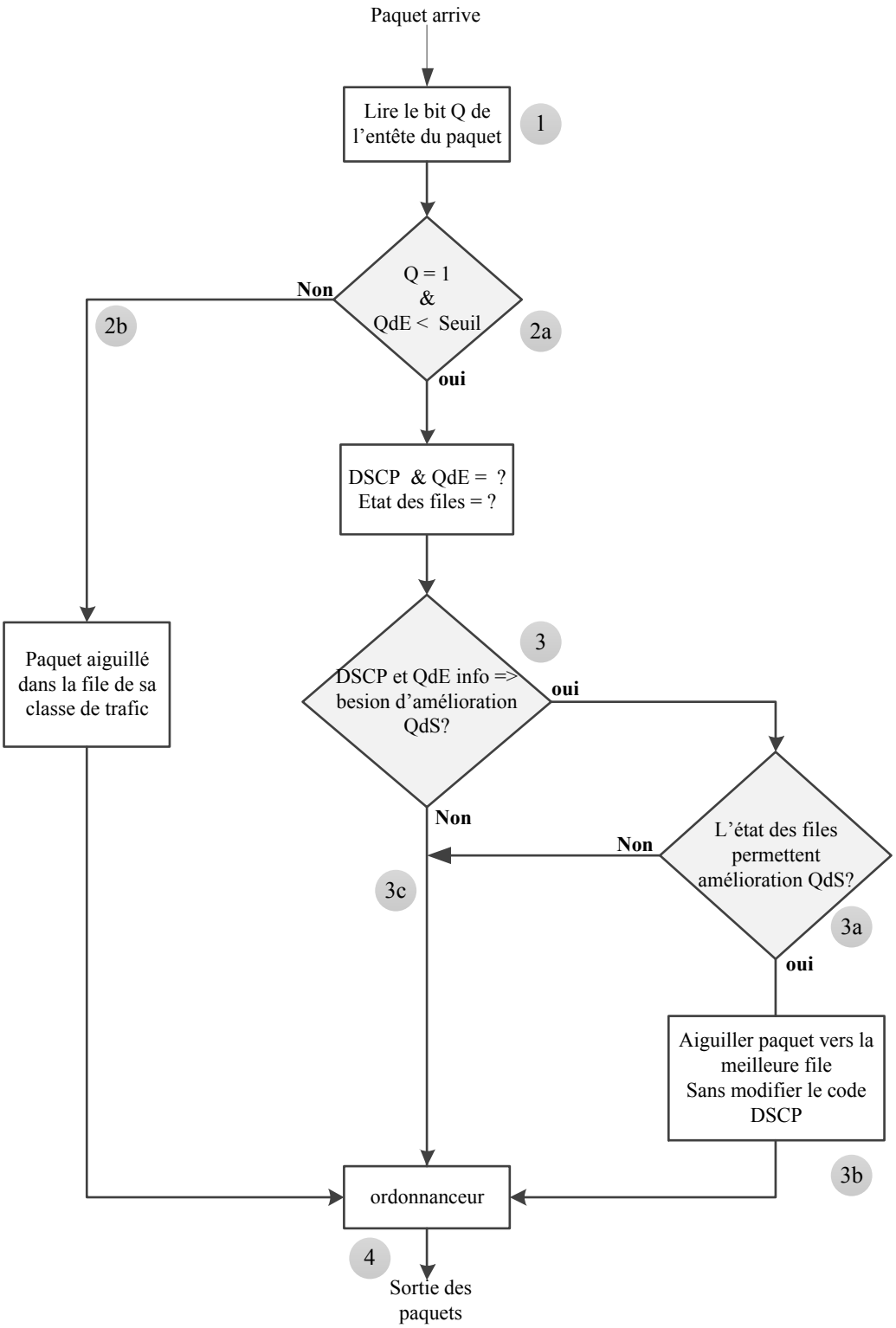


Figure 3.9 Algorithme de traitement des paquets dans le mécanisme proposé

CHAPITRE 4

EXPÉRIMENTATIONS ET RÉSULTATS

Le but d'évaluer la proposition est de voir si elle permet d'améliorer la qualité de l'expérience (QdE) pour les trafics qui y sont sensibles et son effet sur les autres trafics. Bien que la proposition s'étende à tout type de trafic, son évaluation a été faite en considérant le trafic VoIP. Pour évaluer la QdE pour l'application VoIP l'algorithme modèle E (*E-model*) a été utilisé car il n'utilise que des paramètres réseaux objectifs. Cela permet d'avoir facilement dans les simulation l'évaluation de la QdE de l'application VoIP. Le simulateur NS-3 a été utilisé pour réaliser les simulations. Dans les sections qui suivent le simulateur sera présenté suivi de la description de l'algorithme modèle E. Ensuite l'implémentation sera discutée et enfin les résultats de simulations seront analysés.

4.1 Description du simulateur utilisé

L'évaluation de la performance de la proposition détaillée dans le chapitre précédent a été faite en utilisant le simulateur NS-3.13 [43]. NS-3 est un simulateur de réseau à événements discrets destiné principalement aux fins de recherche et d'enseignement. C'est un projet *open-source* développé dans le langage C++. Bien que son nom soit similaire, il n'est pas une extension du très populaire simulateur NS-2 cependant il est développé par le même groupe de travail. Il présente les caractéristiques suivantes :

- Son développement est fait de façon modulaire de façon à faciliter l'intégration de nouveaux modèles
- Il dispose d'un noyau extensible programmé en C++ avec des interfaces optionnelles en python et d'un API très bien documenté. Ce qui permet d'éviter certains problèmes retrouvés dans NS-2 tels que les problèmes de couplage et d'interopérabilités entre les modèles et l'absence de gestion de la mémoire.
- La conception des différents modules dans NS-3 est réalisée avec beaucoup de réalisme et les modules s'alignent sur les systèmes réels. Les modèles de nœuds par exemple sont comme des vrais ordinateurs. Un nœud NS-3 est une module auquel sont ajoutées les applications, les piles de protocole et des interface réseau (*Network Interface Card*).
- Il fournit du support pour la virtualisation et l'intégration aux *testbeds*. Des machines virtuelles peuvent être utilisés au dessus de la couche physique de NS-3. De plus la pile

NS-3 peut être exécutée en mode émulation et émettre/recevoir des paquets à travers des dispositifs réels.

– Les statistiques et les traces d'exécution sont flexibles. Il est capable de générer des fichiers de traces *Packet Capture* (PCAP) qui peuvent être lus avec un analyseur de protocole tel que wireshark [44].
– Les attributs systèmes sont très bien documentés.

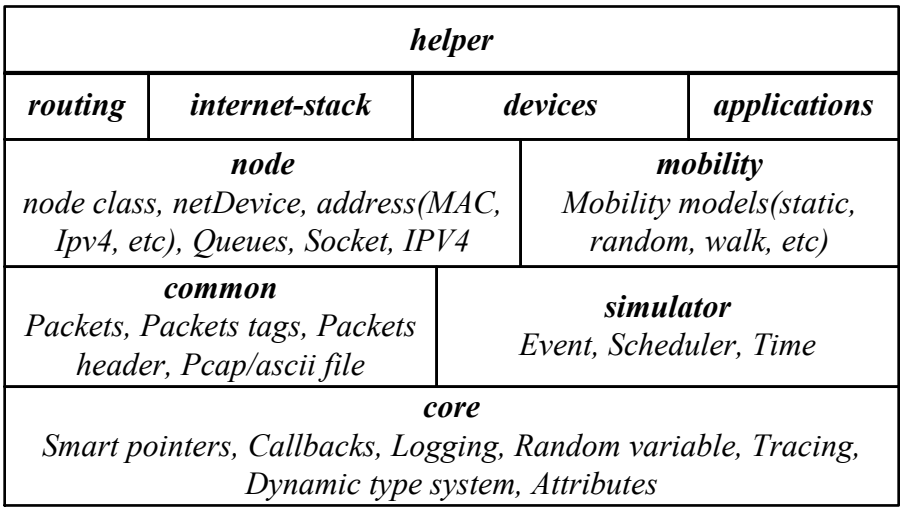


Figure 4.1 Architecture logicielle de NS-3 inspiré de [43]

Le simulateur NS-3 est organisé suivant une structure modulaire et en couche comme présentée à la Figure 4.1 facilitant ainsi son extension. Partant du bas, on retrouve le cœur du simulateur constitué d'un ensemble de composantes qui sont utilisées par toutes les autres classes. On y trouve des classes pour les pointeurs intelligents, des types attributs système dynamique, les traces etc. À la couche 2, il y a des composantes qui sont communes à tous les protocoles et modèles parmi lesquelles se trouvent les objets paquet (*packet*), entête de paquet (*packet header*) et étiquette de paquet (*packet tag*). Au dessus de cette couche, les modèles de nœuds et de mobilité sont définis. Ensuite on retrouve les modèles définissant des protocoles de routage, la pile TCP/IP et les applications. La dernière contient un ensemble de classes dites *helper* qui aident à simplifier le travail des utilisateurs lorsque ces derniers écrivent des scripts de simulations.

4.2 Environnement de simulations

Pour réaliser les simulations, le simulateur a été installé sur une machine virtuelle VMware (VMware Workstation 7.1.3) munie d'un processeur de 2.10GHz à laquelle 1 Gigaoctet de mémoire RAM a été allouée. Sur cette machine est installé le système d'exploitation linux distribution ubuntu 10.04. L'édition des codes sources du simulateur ainsi que la programmation des nouvelles fonctionnalités qui y sont ajoutées se font via l'environnement de développement **Eclipse IDE for C/C++ Developers** version 1.4.1 [45]. La compilation et l'exécution se font à partir de l'outil Waf [46]. La machine hôte de la machine virtuelle possède un processeur Intel Pentium T4300 muni de deux cœurs de 2.10GHz chacun et de 4 Gigaoctet de mémoire RAM et sur laquelle le système d'exploitation WINDOWS 7 version Professionnelle est installé.

4.3 Méthode d'évaluation de la qualité de l'expérience

Comme présenté au chapitre 1, diverses méthodes, dépendant du type d'application, ont été standardisés pour évaluer la QdE. Parmi ces méthodes, nous retrouvons le modèle E, décrit dans la recommandation G-707 de l'ITU [7], qui permet d'évaluer la QdE pour les applications de voix sur IP. Faisant partie des méthodes dites de planification de réseaux, le modèle E utilise des paramètres de performance du réseau pour évaluer la QdE. Aussi dans le cadre des simulations de la voix sur IP, il a été utilisé pour évaluer la QdE.

En effet, le modèle E est un modèle quantitatif qui définit un facteur de qualité R appelé *R-score* qui prend en compte les effets du délai, des pertes de paquets, de la gigue etc., sur la QdE de l'usager. Le facteur R est exprimé comme :

$$R = R_0 - I_s - I_d - I_{e-eff} + A \tag{4.1}$$

Où

- R_0 représente en principe le rapport signal sur bruit de base,
- I_s toutes les dégradations qui se produisent plus ou moins en même temps que le signal vocal,
- I_d prend en compte les effets du délais et des échos,
- I_{e-eff} représente les facteurs de dégradation du aux équipements,
- A un facteur avantage qui prend en compte l'attente de l'usager par rapport à la technologie d'accès utilisée.

En supposant que les pertes de paquets se font de manière aléatoire, I_{e-eff} se calcule

comme suit :

$$I_{e-eff} = I_e + (95 - I_e).\frac{P_{pl}}{P_{pl} + B_{pl}} \tag{4.2}$$

Où

– I_e est le facteur de dégradation dû aux codecs à bas débit,

– P_{pl} taux de perte de paquets,

– B_{pl} représente le facteur de robustesse à l'erreur du codec.

Une approximation de I_d est donnée par l'expression suivante :

$$I_d = 0.24d + 0.11 * (d - 177.3) * I(d - 177.3) \tag{4.3}$$

Avec

$$I(x) = \begin{cases} 0 & \text{si x} < 0 \\ 1 & \text{si x}\geq 0 \end{cases} \tag{4.4}$$

et d, le délai

Dans les relations précédentes, aucune mention explicite n'est faite à la gigue car elle est compensée par l'utilisation d'un tampon à la réception, ce qui induit un délai additionnel. Ce délai dû à la gigue se traduit en une perte pour les paquets qui arrivent après un certain seuil qui dépend de la taille et du type du tampon de gigue utilisé.

Une fois le facteur R déterminé, l'algorithme modèle E établit la correspondance entre R et MOS par la relation suivante :

$$MOS = \begin{cases} 1 & \text{si R} < 0 \\ 1 + 0.035R + R(R - 60)(100 - R)7 * 10^{-6} & \text{si 0 <R<100} \\ 4,5 & \text{si R} \geq 100 \end{cases} \tag{4.5}$$

Les considérations suivantes sont faites dans l'évaluation du MOS. Les délais au niveau du codeur et du décodeur et les échos sont nuls. Pour les échos, il est supposé que des compensateurs d'écho sont utilisés. Les paramètres R_0, I_s , I_e et B_{pl} sont constant et dépendent du codec utilisé.

Tableau 4.1 Mise en correspondance entre les valeurs de R et l'estimation du MOS

Niveau de satisfaction	Facteur R (borne inférieure)	MOS (borne inférieure)
Très satisfait	90	4,34
Satisfait	80	4,03
Quelques utilisateurs insatisfaits	70	3,60
Beaucoup d'utilisateurs insatisfaits	60	3,10
Presque tous les utilisateurs insatisfaits	50	2,58

4.4 Évaluation du MOS dans le simulateur

Le calcul du MOS se fait via un estimateur qui implémente le modèle E et qui dispose de la structure représentée à la Figure 4.2 pour sauvegarder des statistiques sur chaque flot dont la QdE doit être évaluée. Pour chaque flot identifié par le quintuplet (adresse source, adresse destination, port source, port destination, protocole de transport), une fenêtre de temps a été considérée pendant laquelle l'estimateur recueille des statistiques qui le permettent d'évaluer le MOS. La fenêtre de temps considérée est égale à l'intervalle qui a été défini pour l'évaluation de la QdE. Les statistiques recueillies sont le délai, les pertes, la gigue, la quantité de paquets reçus et envoyés.

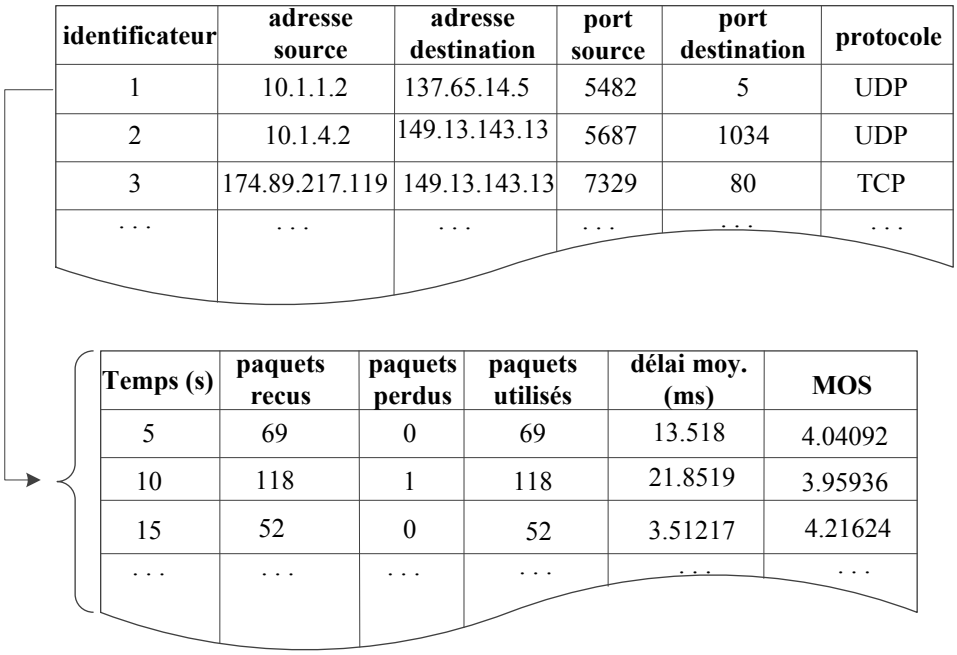


Figure 4.2 Statistique par flot

Le délai et la gigue sont calculées d'après les spécification du protocole RTP [24]. Si S_i représente le temps où le paquet i a été envoyé, et R_i temps d'arrivée du paquet i, alors pour deux paquets i et j le délai D est exprimé comme suit :

$$D(i,j) = (R_j - S_j) - (R_i - S_i) \tag{4.6}$$

Et la gigue associée au paquet i est calculée par la formule suivante :

$$J(i) = J(i-1) + \frac{(|D(i-1,i)| - J(i-1))}{16} \tag{4.7}$$

Avec $J(i-1)$ la gigue pour le paquet précédent et $D(i, i-1)$ le délai entre le paquet i et le paquet $i-1$. Pour les pertes de paquets, ont été comptabilisés tous les paquets qui sont rejetés à n'importe quel endroit du réseau.

4.5 Implémentation des mécanismes

Les simulations ont été en utilisant la version NS-3.13 dans laquelle le modèle DiffServ n'est pas encore été intégré. Cependant l'extension proposée dans [47] a été utilisee en l'intégrant au simulateur. Cette extension implémente un modèle DiffServ avec ses principales fonctionnalités. Les algorithmes *token bucket* décrit à la Figure 2.2, *three rate two colour marking* (trTCM) représenté à la Figure 2.6 et *single rate two colour marking* srTCM décrit à la Figure 2.5) sont utilisés pour faire du contrôle de trafic et du marquage. L'algorithme WRED est utilisé pour la gestion des files d'attente pour les classes AF. Les files EF et DF sont gérées avec l'algorithme *DropTail*. Au niveau de l'ordonnanceur, les algorithmes *Weighted Round Robin* (WRR) (Figure 2.8) et *Rate-controlled Priority Queuing* sont implémentés. Certaines modifications ont été apportées à ce modèle pour que le traitement des paquets prennent en compte l'information de la QdE. Le diagramme de la Figure 4.3 montre les principales classes qui ont ajoutées au simulateur.

Pour qu'un émetteur ait accès aux informations de la QdE évaluées au récepteur, il va les lire dans la structure de données dont un exemple est fourni à la Figure 4.2. A travers une étiquette (*label*) que NS-3 permet de 'coller' aux paquets, l'émetteur introduit dans le réseau les informations de la QdE qu'il a lues. L'étiquette ajoutée au paquet joue le rôle de l'entête ConEx puisque Re-ECN n'est pas encore implémenté. Lorsqu'un paquet arrive à un routeur, ce dernier accède à l'étiquette pour y lire les informations de la QdE. Ainsi suivant la valeur du MOS et la tendance dans la variation du MOS, le paquet est aiguillé vers la meilleure file d'attente.

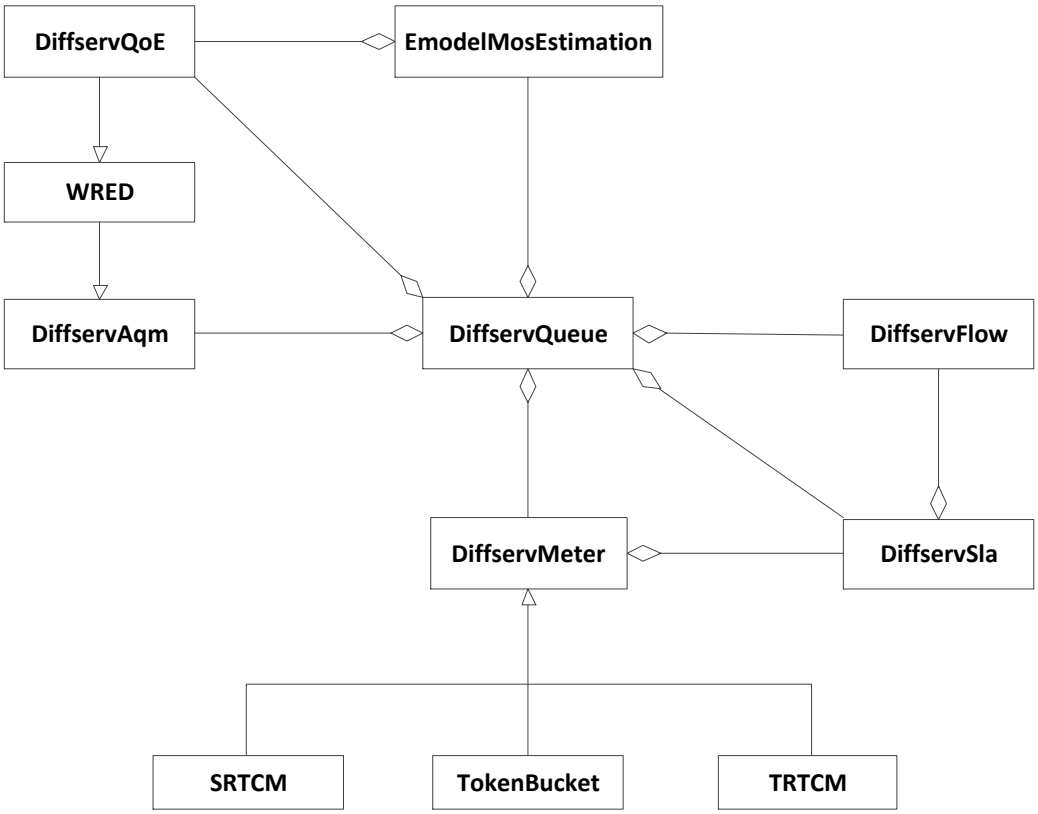


Figure 4.3 Digramme des principales classes ajoutées au simulateur

La tendance de la variation de l'information de la QdE représentée par l'estimation du MOS est déterminée par une droite de régression qui est calculée à partir d'un fenêtre de valeurs du MOS. Cette fenêtre est configurable. La pente de cette droite renseigne sur la tendance de la variation des valeurs du MOS.

Ces opérations ont été réalisées par deux classes principales que nous avons ajoutées au simulateur.

La classe **EmodelMosEstimation** dont l'attribut principal est la structure de stockage présentée à la Figure 4.2 implémente l'estimateur de la QdE qui effectue tous les traitements devant aboutir à l'évaluation de la QoE. Les principales méthodes sont :

– *PacketSent (Ptr<Node>)* : Cette méthode s'applique à un nœud émetteur qui envoie du trafic pour lequel l'information de la QoE doit être évaluée. Pour chaque paquet qui est en train d'être envoyé par ce nœud, la méthode assigne un identifiant correspondant au quintuplet (voir Figure 4.2) qui identifie le flot dans lequel le paquet appartient. Ensuite cette méthode ajoute au paquet une étiquette munie de cet identifiant de sorte que les

autres nœuds puissent identifier le flot auquel appartient le paquet sans avoir à lire son entête. Enfin, si les dernières informations de la QdE pour le flot sont disponibles, la méthode les rajoute au paquet via une étiquette.

- *PacketReceived (Ptr<Node>)* : Appliquée à un nœud récepteur, cette méthode détermine, pour chaque paquet d'un flot qu'elle reçoit, le délai et la gigue et sauvegarde ces informations jusqu'à qu'elles soient utilisées pour déterminer la QdE. La durée de sauvegarde dépend de l'intervalle de calcul.
- *PacketDrop ()* : Cette méthode agit sur tous les nœuds entre l'expéditeur et le destinataire pour faire le suivi de chaque d'un flot en reportant tous les paquets qui sont perdus ou éliminés par n'importe quel nœud du réseau
- *GetPeriodicFlowStat ()* : Cette méthode calcule à intervalle fixe l'information de la QoE pour chaque flot et reporte l'information dans la structure de stockage de l'estimateur.
- *EmodelEstimation ()* : Méthode implémentant l'algorithme décrit dans la section 4.3 qui calcule le MOS.

D'autre méthodes telles que *TxPacketLogger()*, *RxPacketLogger()*, *DropPacketLogger()*, *QueuePacketLogger()*, *Ipv4PacketLogger()*, *GetSlope()*, *GetPeriodicFlowStat()*, *GetLastQoE()* existent pour trouver la dernière information de la QoE pour un flot donné, estimer le sens de variation de l'information de la QoE et faire sortir les statistiques pour analyses ultérieures, etc.

La classe **DiffServQoE** maintient l'état des files d'attente WRED des routeurs cœur du réseau DiffServ dans des structures de données. Ses principales méthodes sont :

- *PredictedDropProbability ()* : Étant donné un paquet, cette méthode détermine avec quelle probabilité que ce paquet serait éliminé s'il était inséré dans la file qui correspond à sa classe ou dans les autres files de plus grande priorité.
- *GetPredictedDelay ()* : En tenant compte du nombre d'octets présent dans les différentes files et du poids relatif de chaque file, cette méthode estime le délai que passerait un paquet qui vient d'arriver au routeur s'il y était inséré.
- *GetSuitableQueue ()* : Cette méthode permet de déterminer si le paquet qui arrive sera admis dans sa file par défaut ou dans la file supérieure. Cette décision se base sur le délai prédit et la probabilité que le paquet soit éliminé. Avant de prendre la décision, elle vérifie le niveau d'occupation de la file et vérifie si ce niveau d'occupation est inférieure au seuil qui a été défini pour qu'une file d'attente accepte des paquets provenant d'autres classes de service.

La classe**DiffServQueue** qui implémente les traitements effectués par les routeurs à la frontière et à l'intérieur d'un domaine DiffServ a été modifiée notamment en ajoutant la méthode **QoEDeteriorated** qui vérifie, lorsqu'un paquet arrive au routeur, si la QdE s'est

dégradée et, si nécessaire, le faire passer dans une file supérieure.

4.6 Simulation de la voix sur IP

Pour simuler la voix sur IP, le modèle présenté dans [48] a été utilisé. On suppose que le codeur utilisé est le G.711 avec anéantisseur d'échos. Le trafic fourni par le codeur est caractérisé par une succession de période active où il envoie des données (période ON) et de période de silence (période OFF). Pendant la période ON, les paquets sont envoyés a intervalle régulier correspondant au temps de paquétisation a un débit maximal de 29.6 kb/s. Typiquement, pour le codeur utilisé, le temps de paquétisation est estimée a 20ms. Les périodes ON et OFF sont estimées par une loi exponentielle de moyenne de 0.35 sec et 0.65 sec respectivement. Il s'ensuit que la charge utile des paquets VoIP sera de 172 octets (160 octets pour la voix et 12 octets pour l'entête RTP).

4.7 Plan d'expérience

Diverses simulations ont été réalisées pour évaluer la performance des mécanismes proposés. Deux types de scénario ont été considérés. Un scénario simple et un scénario complexe. La métrique principale qui sera évaluée dans les deux scénarios est le MOS. Le scénario simple a permis de voir la variation de la QdE au cours de la simulation avec et sans le mécanisme proposé. Le scénario complexe, de son coté, a permis d'évaluer globalement la QdE de l'application pour les différents cas simulés. Dans tous les types de scénarios 4 types de trafic sont utilisés. Le trafic VoIP pour lequel le mécanisme peut être appliqué, du trafic FTP et du trafic CBR avec UDP. Pour ce qui a trait à l'ordonnancement, l'algorithme WRR a été utilisé et les files AF1, AF2, AF3 et BE ont respectivement un poids de 8/20, 6/20, 4/20 et 2/20. Chaque file peut contenir au plus 100 paquets. Le seuil défini pour le MOS est de 4.0, l'intervalle d'évaluation est fixé à 5 secondes. Rien n'est précisé pour le taux d'occupation des files d'attente.

4.7.1 Scénario simple

Le scénario simple est un scénario peu chargé constitué de 4 clients dont deux font de la voix sur IP. deux classes de trafic AF1 et AF2 ont été considérées. La classe AF2 a été surchargée pour provoquer une dégradation de la QdE du flot VoIP de classe AF2 (voir Table 4.2). Le réseau de la Figure 4.4 a été simulé pour ce scénario. Ce scénario vise essentiellement à montrer la variation du MOS lorsque DiffServ est utilisé seul et lorsque le *feed-back* de la

QdE est utilisé dans la classification.

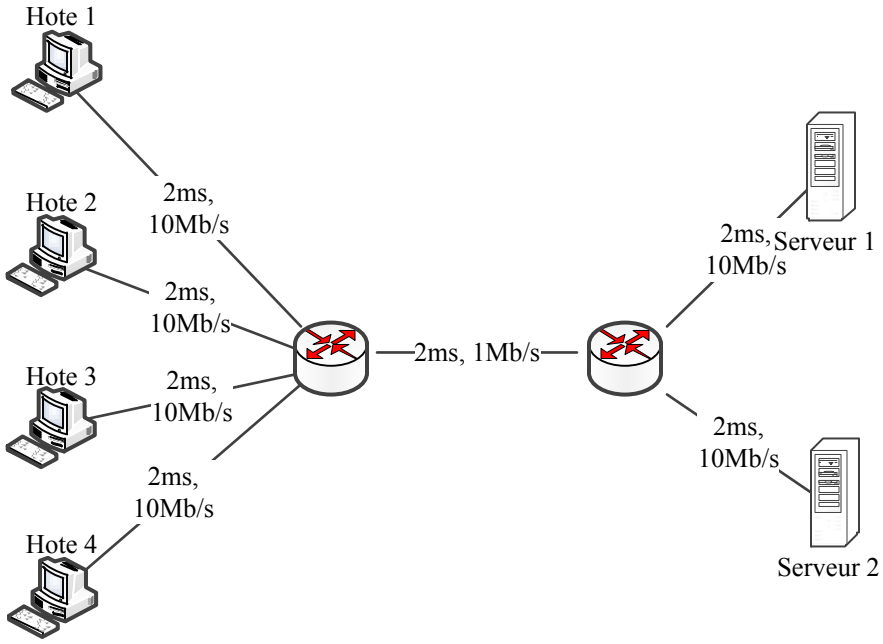


Figure 4.4 Réseau du scénario simple

Tableau 4.2 Flots pour le scénario simple

No Flot	Source->Destination	Application	Classe
1	Hote1 ->Serveur 1	VoIP	AF1
2	Hote2 ->Serveur 2	VoIP	AF2
3	Hote3 ->Serveur 1	Trafic CBR	AF2
4	Hote4 ->Serveur 1	Trafic CBR	AF2

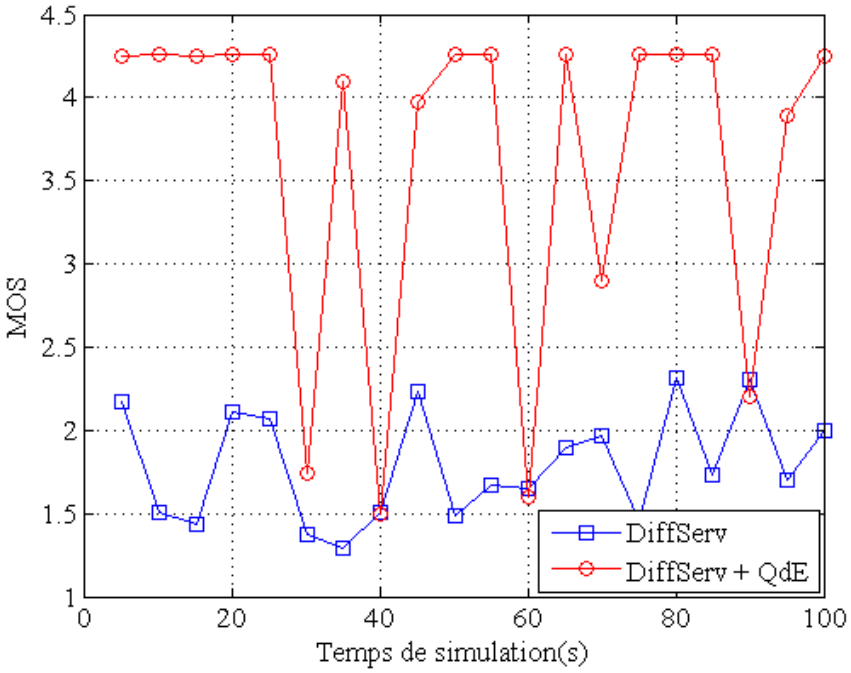


Figure 4.5 Graphe de la variation du MOS pour le flot 2 avec et sans le mécanisme

Les simulations ont été réalisées avec et sans le mécanisme. Dans chaque cas, les variations des métriques du MOS, du délai moyen, et des pertes de paquets moyennes pour chacun des intervalles d'évaluation du MOS ont été inventoriés. Lorsque le mécanisme n'est pas utilisé la QdE du flot de VoIP de classe AF2 exprimée par le MOS varie entre 1.28 et 2.31 avec une moyenne de 1.79. Avec de telles valeurs pour le MOS, la communication VoIP serait quasiment impossible. Lorsque le classification prend en compte la QdE, les valeurs minimale, maximale et moyenne du MOS sont respectivement 1.49, 4.25 et 3.64. Ce qui assure une qualité acceptable pour la communication.

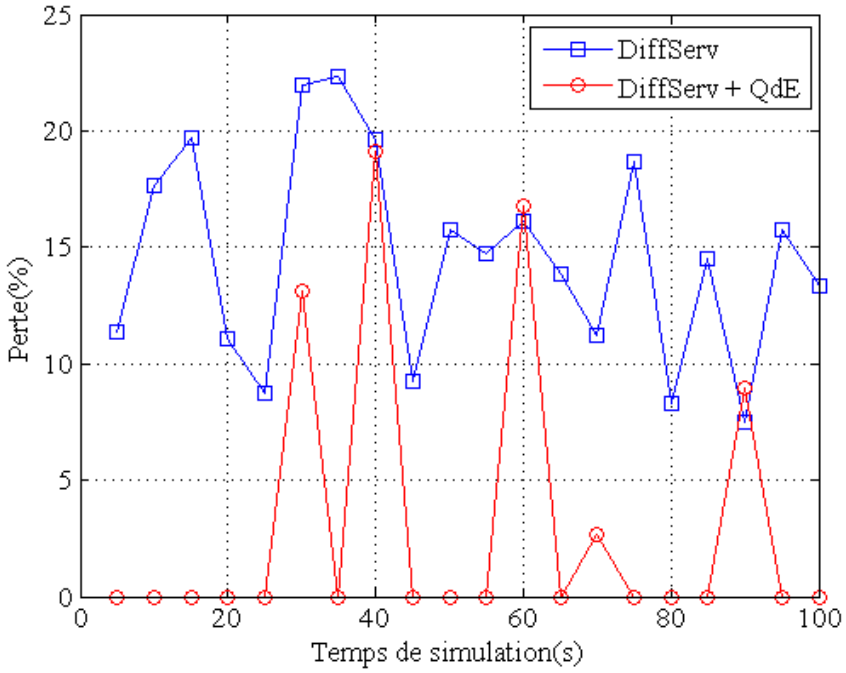


Figure 4.6 Variation des pertes pour le flot 2 avec et sans le mécanisme

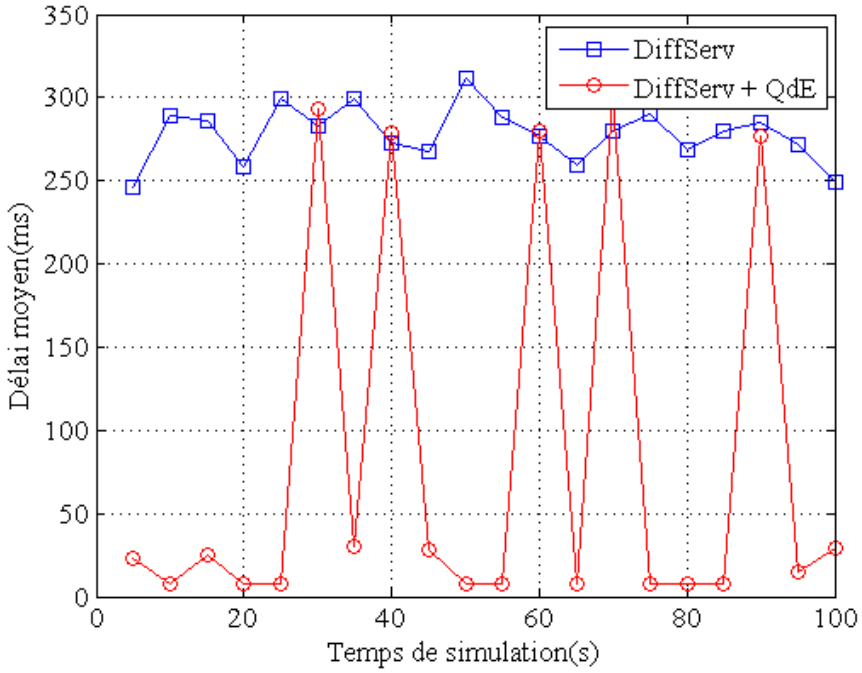


Figure 4.7 Variation du délai pour le flot 2 avec et sans le mécanisme

Il convient aussi de remarquer l'impact du mécanisme sur les autres métriques de performance du réseau. En effet, les graphes des Figures 4.6 et 4.7 montrent l'impact du mécanisme sur la perte de paquets ainsi que sur le délai moyen expérimenté par les paquets.

4.7.2 Scénario complexe

Ce type de scénario se base sur l'architecture de réseau représentée à la Figure 4.8. Le réseau est constitué de 24 sources dont 10 font de la voix sur IP à un débit moyen de 28kbps répartis entre les classes AF1 et AF2, 10 font du FTP avec des flots répartis entre les classes AF2 et AF3 et les 4 autres envoient du trafic à débit constant (trafic CBR) de classe BE. Dix (10) scénarios ont été simulés et dans chacun des cas le nombre de flots de VoIP ont été gardés constant tandis que la quantité des autres flots a augmenté. Le Tableau 4.3 spécifie pour chaque scénario le nombre de flots et la proportion de trafic pour l'application VoIP.

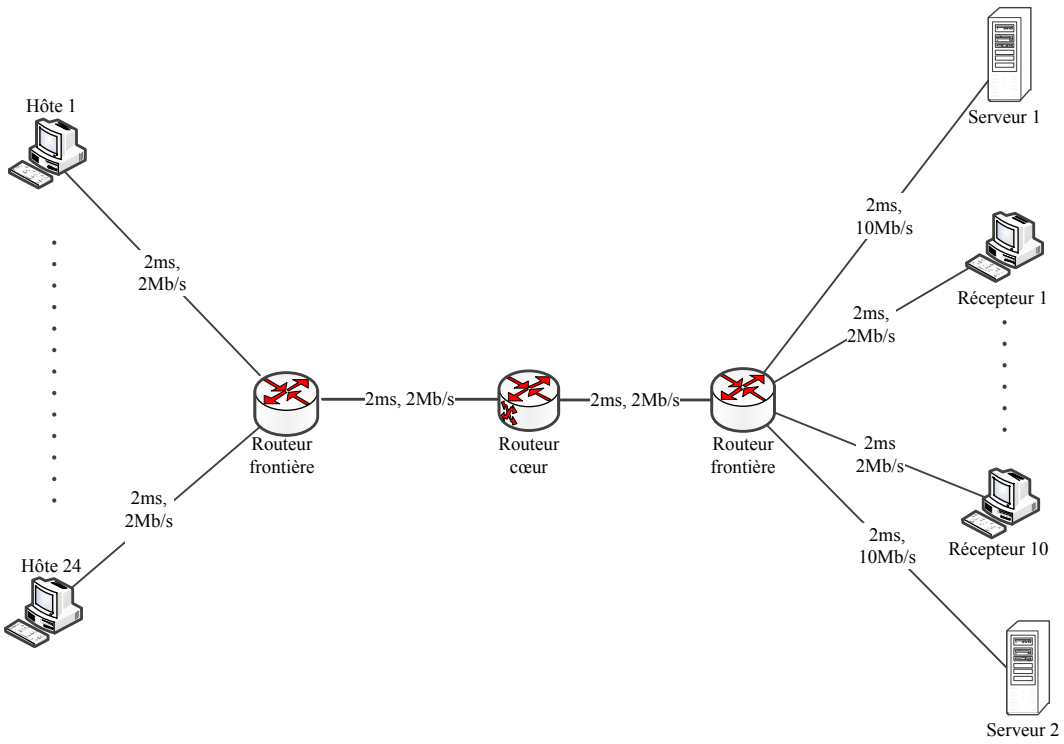


Figure 4.8 Réseau du scénario complexe

Les Tableaux 4.5 et 4.4 présentent les résultats et comparent le cas où le mécanisme n'est pas utilisé versus lorsqu'il est utilisé. Le Tableau 4.4 montre les valeurs du MOS mesurés par scénario. La valeur moyenne concerne toutes les communications tandis que la la valeur minimale indique la valeur du MOS pour la communication qui a la QdE relative la plus faible

Tableau 4.3 Scénarios simulés

Scénario	Nombre de flots	% Trafic VoIP
1	10	100
2	12	83.33
3	13	79.92
4	15	66.66
5	16	62.5
6	18	55.55
7	19	52.63
8	21	47.61
9	22	45.45
10	24	41.66

Tableau 4.4 Résultats pour les différents scénarios avec et sans le mécanisme

Scénario	Moyenne(MOS)		Min(MOS)		Max (MOS)	
	DiffServ	DiffServ & QoE	DiffServ	DiffServ & QoE	DiffServ	DiffServ & QoE
1	4.23	4.23	4.04	4.04	4.25	4.25
2	4.20	4.24	4.16	4.21	4.26	4.25
3	4.14	4.20	3.98	4.11	4.25	4.25
4	4.08	4.16	3.8	4.04	4.25	4.25
5	3.59	3.81	3.06	3.64	4.03	3.99
6	3.65	3.72	3.16	3.43	4.05	3.94
7	3.63	3.74	3.10	3.48	4.05	3.95
8	3.41	3.57	3.17	3.29	3.73	3.78
9	3.36	3.50	3.04	3.37	3.63	3.59
10	3.27	3.50	2.86	3.35	3.57	3.67

et la valeur maximale indique la valeur du MOS pour la communication ayant la QdE relative la plus élevée des dix communications VoIP. Pour les scénarios 1 et 2, l'effet du mécanisme n'est pas important puisque la valeur du MOS est supérieur à 4 avec ou sans le mécanisme proposé. En revanche les scénarios subséquents montrent l'effet du mécanisme notamment à partir du scénario 4. Le Tableau 4.5 montre le gain obtenu pour les différents scénarios.

Les graphes des Figures 4.9, 4.11, 4.12, 4.13 et 4.14 suivantes comparent respectivement le MOS pour les trafic VoIP de la classe AF2, le MOS, le délai moyen, les pertes moyennes et la gigue pour tous flots VoIP en fonction de ce que représente le trafic VoIP du trafic total dans le cas où le mécanisme proposé est utilisé versus lorsqu'il n'est pas utilisé. L'amélioration globale qui se fait, comme le montre le graphe de la Figure 4.11, est due à l'amélioration de la QdE des trafics VoIP de la classe AF2 (voir le graphe de la Figure 4.9). Le graphe de la

Tableau 4.5 Gain par rapport au cas où le mécanisme n'es pas utilisé

Scénario	%Min(MOS)	% Max(MOS)	%Moyenne
1	0	0	0
2	1.18	- 0.23	0.23
3	3.16	0	1.42
4	5.94	0	1.92
5	15.93	-1	5.77
6	7.87	-2.79	1.88
7	10.91	-2.53	2.94
8	3.64	1.32	4.48
9	9.79	-1.11	4
10	14.62	2.72	6.83

Figure 4.10 affiche le délai moyen pour les flots AF2 dans les deux cas. Lorsque le mécanisme est utilisé le délai maximal expérimenté par les paquets est de 258 ms avec une moyenne de 173 ms et lorsqu'il n'est pas utilisé, le délai max est de 269 ms avec une moyenne de 173 ms.

La Figure 4.15 montre le peu d'impact qu'à le mécanisme proposé sur la qualité des communications de la classe AF1 autrement dit le mécanisme utilisé pour faire passer certains paquets de AF2 dans AF1 n'affecte pas la qualité qu'auraient dû avoir les trafics de classe AF1.

Le graphique à bâtons de la Figure 4.16 compare le débit des trafics d'arrière plan avec le mécanisme versus sans le mécanisme. Les résultats montrent que le débit a très peu varié dans les deux cas.

Tout compte fait les résultats montrent que le fait de prendre en compte l'information du *feed-back* de la qualité d'expérience a une incidence sur les performances globales du réseau et permet par exemple, comme le montre la Figure 4.13, de diminuer dans ce cas le taux de perte de paquets. Comme il en sera mention dans la section 5.3 du chapitre suivant, il demeure encore possible d'avoir de meilleurs résultats en jouant sur les paramètres tels que l'intervalle d'évaluation de la QdE, le niveau d'occupation d'une file pour qu'elle accepte à un instant donné du trafic d'autres classes et le seuil défini pour le MOS.

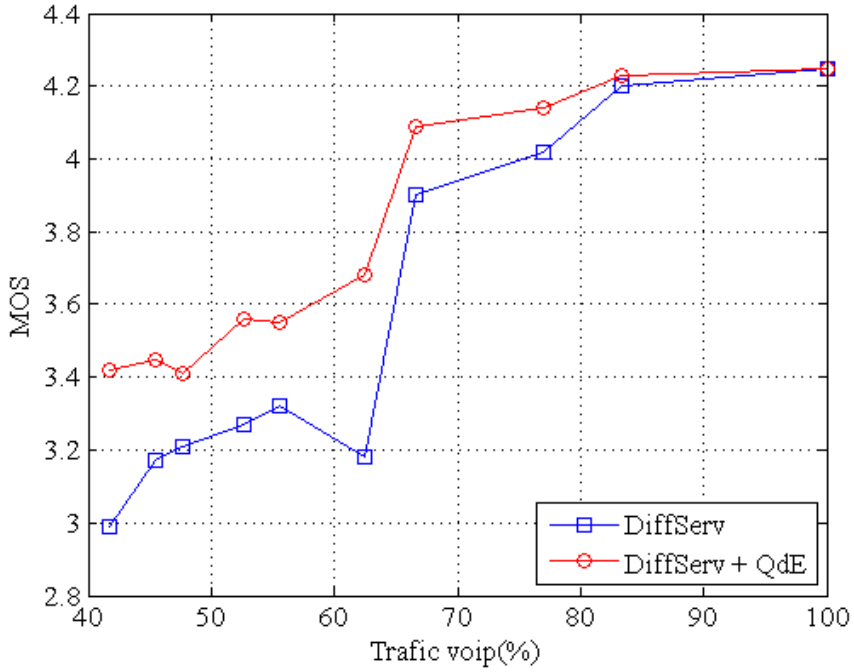


Figure 4.9 Graphe du MOS pour les flots AF2

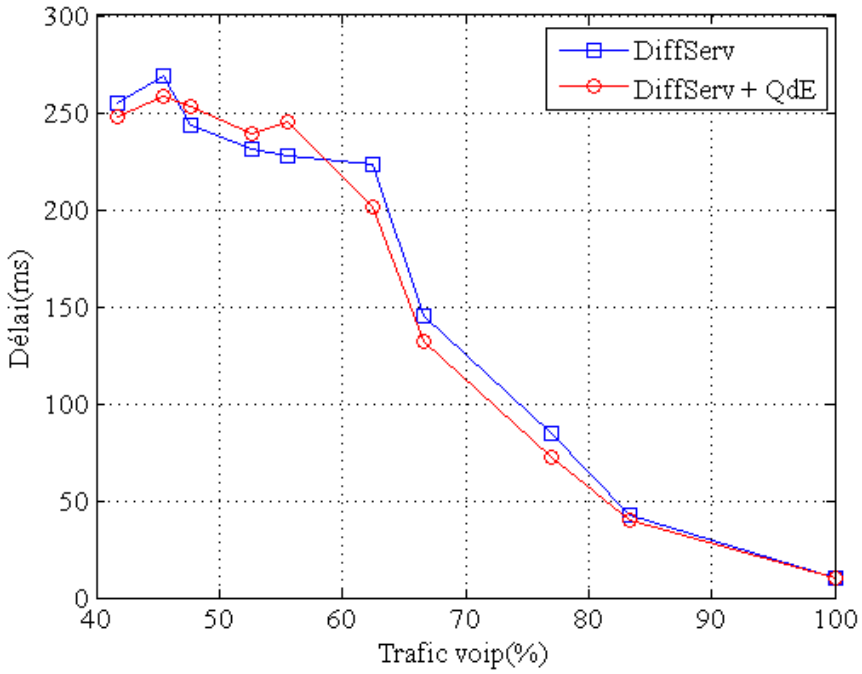


Figure 4.10 Délai moyen des flots AF2

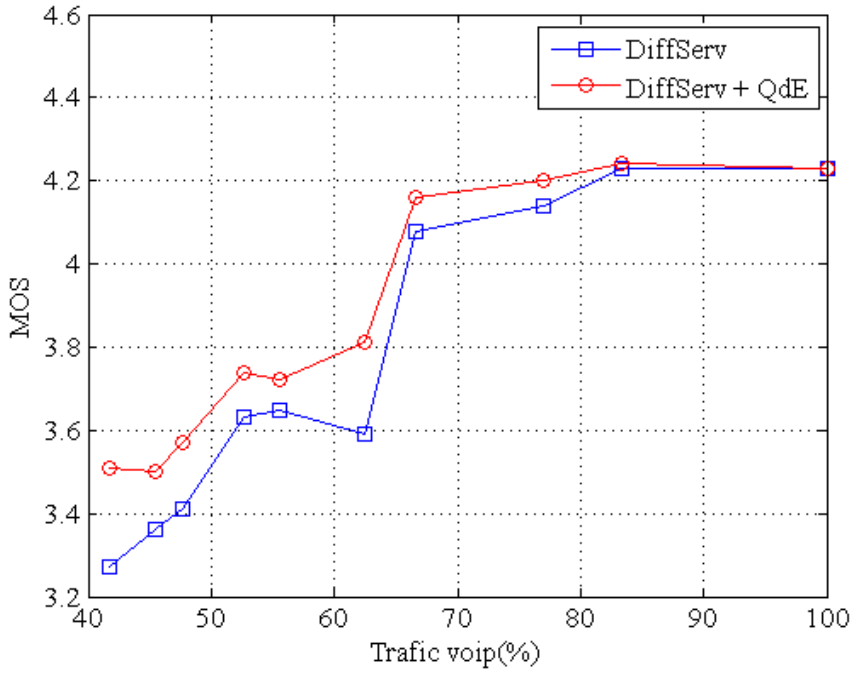


Figure 4.11 Graphe de la moyenne du MOS pour tous les flots VoIP

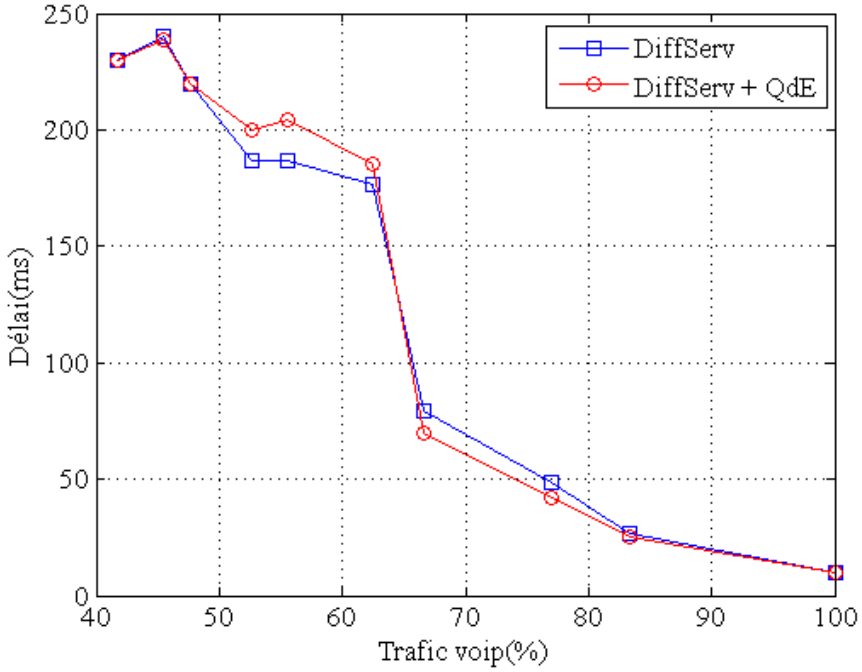


Figure 4.12 Graphe du délai moyen tous les flots VoIP

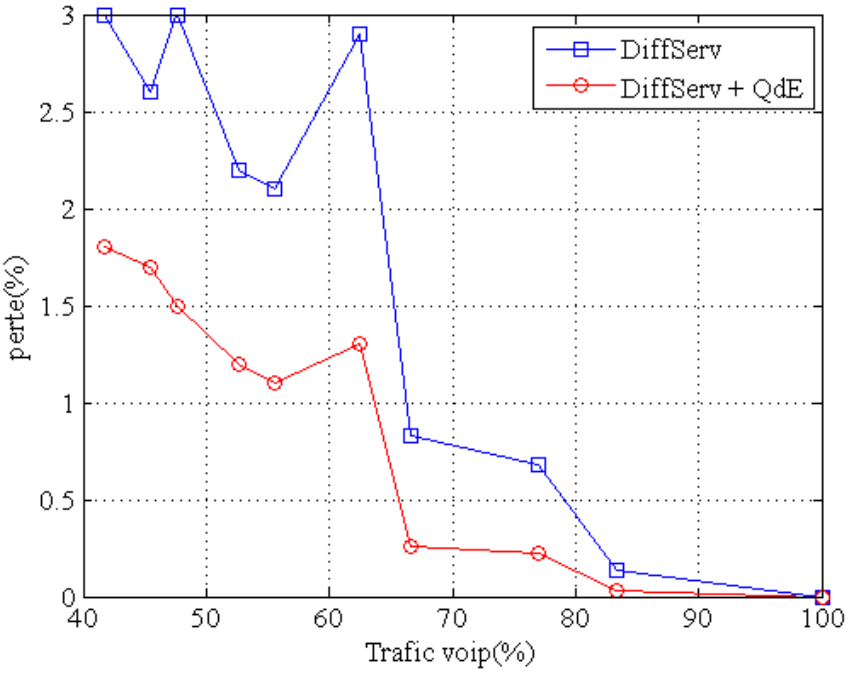


Figure 4.13 Graphe des pertes moyennes tous les flots VoIP

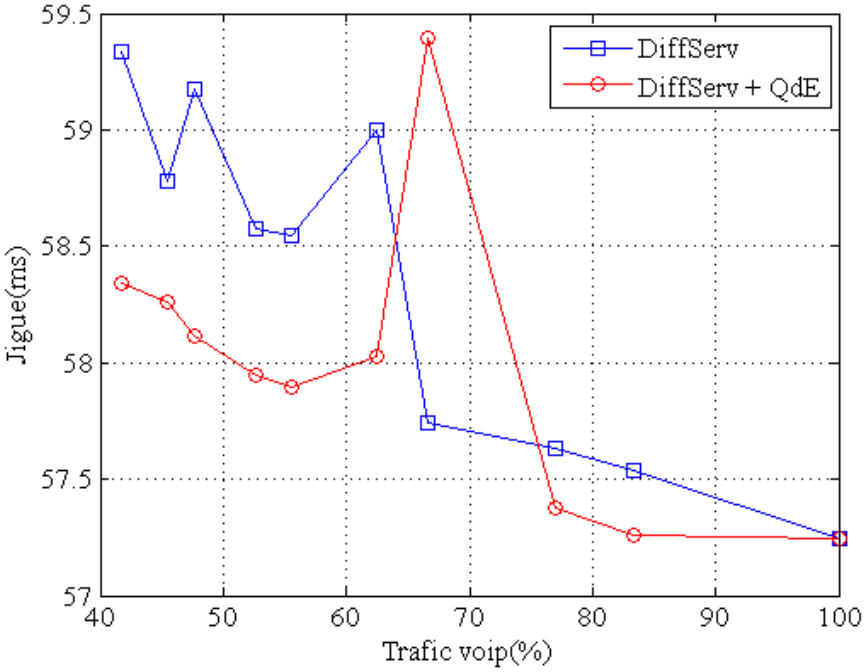


Figure 4.14 Graphe de la gigue moyenne des flots VoIP

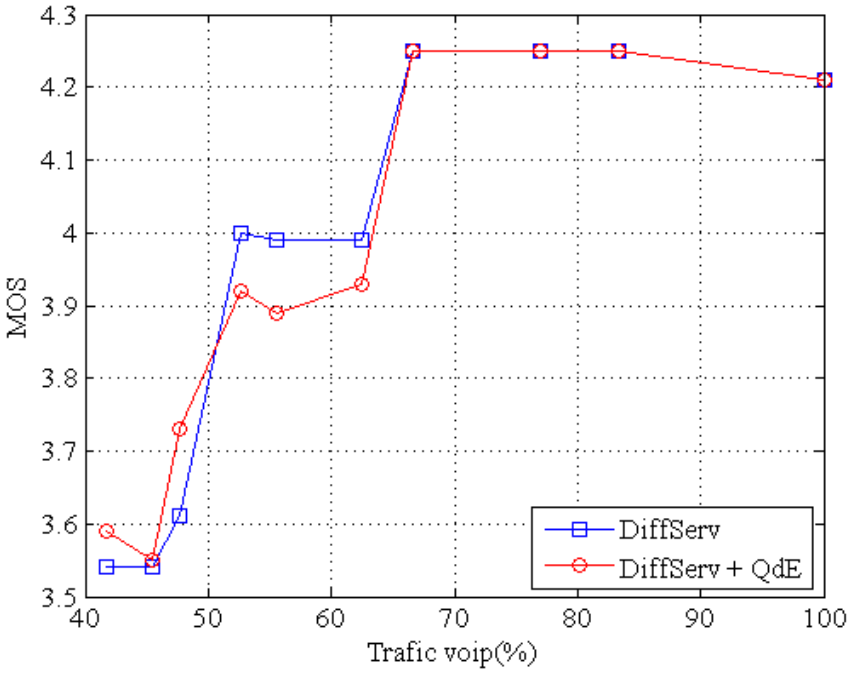


Figure 4.15 Graphe de la moyenne du MOS pour tous les flots AF1

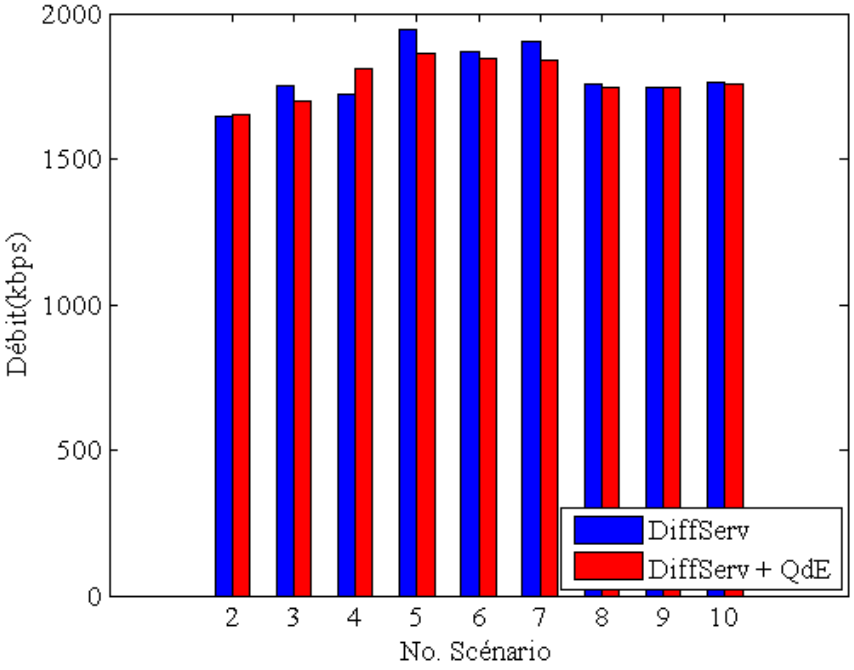


Figure 4.16 Graphique du debit des trafics d'arrière-plan

CHAPITRE 5

CONCLUSION

Dans le cadre de cette maîtrise, nous avons développé un nouveau mécanisme qui permet d'améliorer le niveau de satisfaction des usagers des services dont la qualité de l'expérience (QdE) peut être déterminée et retransmise dans le réseau. Dans les sections suivantes, une synthèse du travail sera effectuée ensuite, nous ferons ressortir ses limitations et enfin nous fournirons quelques pistes pour des améliorations futures.

5.1 Synthèse des travaux

Dans ce mémoire, nous avons proposé un nouveau type de traitement de paquets qui prend en compte l'information de la QdE, lorsqu'elle est disponible, lors de la classification des paquets faite par les routeurs internes à un domaine DiffServ. Ce nouveau mode de traitement des paquets n'est que l'une des possibilités de l'utilisation du mécanisme de *feed-back* que nous avons également proposé pour faire la rétroaction de l'information de la QdE. Pour permettre aux routeurs d'avoir accès à l'information de la QdE, l'émetteur l'écrit dans l'entête des paquets lorsque le protocole de la couche 3 est IPv6 soit en utilisant les bits restants de l'entête d'option destination du protocole ConEx ou en créant une autre entête de destination. Dans le cas de IPv4, vu qu'il ne reste qu'un seul bit, l'émetteur ne fait qu'indiquer par ce bit si la QdE s'est dégradée ou non. Pour évaluer les performances des mécanismes proposés, nous les avons simulés avec le simulateur NS-3.13 auquel nous avons intégré un module DiffServ que nous avons modifié. Les simulations effectuées, en utilisant du trafic VoIP, montrent que les mécanismes proposés permettent d'améliorer le niveau de la QdE pour les flots VoIP et, en même temps, améliorer les métriques de qualité de service.

5.2 Limitations des travaux

Les mécanismes proposés souffrent de quelques limitations. Le mécanisme de *feed-back* proposé ne pourra pas cohabiter avec Re-ECN si ce dernier est adopté par l'IETF pour les applications utilisant le protocole de transport TCP sur IPv4, car le seul bit qui permettrait de signaler la dégradation de la QdE aux routeurs est utilisé par le protocole Re-ECN. L'application faite à DiffServ, via le nouveau mécanisme de classification des paquets aux

routeurs internes à un domaine DiffServ, peut conduire, dans certains cas, à perdre la séquence des paquets, ce qui nécessitera d'avoir des mécanismes de ré-ordonnancement à la réception.

Pour ce qui concerne les simulations, ConEx/Re-ECN n'est pas implémenté pour transmettre l'information de la QdE dans le réseau. Les informations de la QdE pour tous les flots sont stockées dans une seule structure dans laquelle les nœuds récepteurs et émetteurs écrivent et lisent respectivement les informations relatives à la QdE. Le fait, pour les émetteurs, d'aller lire directement l'information de la QdE, la rend disponible pour ces derniers plus rapidement qu'elle ne le serait dans la réalité. Dans la réalité, il faudra au moins la moitié d'un *Round Trip Time* (RTT) avant que l'émetteur ne sache la dernière information de QdE estimée au récepteur. D'un autre coté, l'application VoIP avec sa pile protocolaire n'est pas encore implémentée dans la version du simulateur utilisé. Ainsi pour la simuler, nous avons utilisé un modèle théorique qui est fourni dans [48]. En outre, dans l'évaluation de la QdE par le modèle E (*E-model*), nous avons supposé que les pertes des paquets dans le réseau se font de façon aléatoire, mais en réalité, pour l'application VoIP, les pertes de paquet se produisent en rafale, autrement dit, à un instant donné, il peut y avoir beaucoup de pertes de paquets et aucune l'instant d'après. Il résulte de ces considérations que les résultats obtenus ne sont que des approximations.

5.3 Travaux futurs

Plusieurs autres travaux peuvent découler des mécanisme proposés en guise de travaux futurs. Des améliorations peuvent être apportées à l'application faite à DiffServ en utilisant l'information de la QoE au niveau des traitements effectués par les routeurs situés à la frontière d'un domaine DiffServ dans le contrôle de trafic, par exemple. D'autre part, il serait intéressant d'avoir un mécanisme de contrôle d'accès basé sur la QoE de façon à bloquer certaines communications si elles vont détériorer la QdE, ou si l'état du réseau laisse prévoir pour ces dernières, que les contraintes de QdE ne pourront pas être satisfaites. De plus, des mécanismes de sécurité seront nécessaires pour s'assurer que des nœuds ne trichent pas en altérant la QdE évaluée au récepteur. Enfin, le *feed-back* de la QdE peut être utilisé dans d'autres mécanismes d'*active queue management* comme codel [49] ou d'autres mécanismes de QdS.

Concernant les résultats, l'utilisation d'un *testbed* dans lequel de vraies communications VoIP seraient effectuées permettrait d'évaluer avec d'avantage de précisions l'impact du nouveau mécanisme sur les communications. Pour avoir des résultats reflétant d'avantage la réalité, il serait intéressant d'utiliser les protocoles qui ont été mentionnés pour avoir le *feed-back* de l'information de la QdE et l'insérer dans le réseau. Pour une meilleure évaluation

de la QdE par le modèle de E, le modèle de Markov trouvé dans [50] peut être utilisé pour prendre en compte la caractéristique en rafale des pertes de paquets.

RÉFÉRENCES

[1] ITU-T, *P. 800 : Methods for subjective determination of transmission quality*, Std., 1996.

[2] H. Tran et A. Mellouk, "Qoe model driven for network services," *Wired/Wireless Internet Communications*, 264–277, 2010.

[3] ETSI, "Human factors ; quality of experience (qoe) requirements or real-time communication services," ETSI, France, Rapport technique, 2009.

[4] E. Crawley et R. Nair, "A framework for qos-based routing in the internet," RFC2386, Août 1998. [En ligne]. Disponible : http://www.ietf.org/rfc/rfc2386.txt

[5] H. Rifai, S. Mohammed, et A. Mellouk, "A brief synthesis of qos-qoe methodologies," *2011 10th International Symposium on Programming and Systems.* Piscataway, NJ, USA : IEEE, 2011, 32–8, 2011 10th International Symposium on Programming and Systems, 25-27 April 2011.

[6] R. Serral-Gracià, E. Cerqueira, M. Curado, M. Yannuzzi, E. Monteiro, et X. Masip-Bruin, "An overview of quality of experience measurement challenges for video applications in ip networks," *Wired/Wireless Internet Communications*, 252–263, 2010.

[7] ITU-T, "G. 107 : The e model, a computational model for use in transmission planning," *Intl. Telecom. Union*, 2005.

[8] S. Khirman et P. Henriksen, "Relationship between quality-of-service and quality-of-experience for public internet service," 2002.

[9] E. Ekudden, R. Hagen, I. Johansson, et J. Svedberg, "The adaptive multi-rate speech coder." IEEE, 1999, 117–119.

[10] T. Hai Anh, A. Mellouk, et S. Hoceini, "User to user adaptive routing based on qoe," *2011 10th International Symposium on Programming and Systems.* Piscataway, NJ, USA : IEEE, 2011, 39–45, 2011 10th International Symposium on Programming and Systems, 25-27 April 2011.

[11] S. Floyd, R. Gummadi, et S. Shenker, "Adaptive red : An algorithm for increasing the robustness of red active queue management," 2001.

[12] C. Semeria, "Supporting differentiated service classes : queue scheduling disciplines," *Juniper networks*, 2001.

[13] A. Capone, L. Fratta, et F. Martignon, "Dynamic online qos routing schemes : Performance and bounds," *Computer Networks*, vol. 50, no. 7, 966–981, 2006.

[14] S. Sahu, P. Nain, C. Diot, V. Firoiu, et D. Towsley, "On achievable service differentiation with token bucket marking for tcp," vol. 28. ACM, 2000, 23–33.

[15] R. Braden, D. Clark, et S. Shenker, "Integrated services in the internet architecture," RFC 1633, Juin 1994. [En ligne]. Disponible : http://tools.ietf.org/html/rfc1633

[16] R. Braden, L. Zhang, S. Berson, S. Herzog, et S. Jamin, "Resource reservation protocol (rsvp)," RFC 2205, IETF, Rapport technique, 1997.

[17] J. Wroclawski, "The use of rsvp with ietf integrated services," RFC 2210, 1997. [En ligne]. Disponible : http://www.ietf.org/rfc/rfc2210.txt

[18] S. Black, D. Carlson, M. Davies, E. Wang, et Z. Weiss, "An architecture for differentiated services," RFC 2475, Décembre 1998. [En ligne]. Disponible : http://www.ietf.org/rfc/rfc2475.txt

[19] J. Heinanen et R. Guerin, "A single rate three color marker," RFC 2697, 1999. [En ligne]. Disponible : http://tools.ietf.org/html/rfc2697

[20] J. Heinanen et R. Guerin, "A two rate three color marker," RFC 2698, 1999. [En ligne]. Disponible : http://tools.ietf.org/html/rfc2698

[21] V. Jacobson, K. Nichols, et K. Poduri, "An expedited forwarding phb," RFC 2598, Juin 1999. [En ligne]. Disponible : http://www.ietf.org/rfc/rfc2598.txt

[22] J. Heinanen, F. Baker, W. Weiss, et J. Wroclawski, "Assured forwarding phb group," RFC 2597, Juin 1999. [En ligne]. Disponible : http://www.ietf.org/rfc/rfc2597.txt

[23] K. Ramakrishnan, S. Floyd, D. Black *et al.*, "The addition of explicit congestion notification (ecn) to ip," RFC 3168, 2001. [En ligne]. Disponible : http://www.ietf.org/rfc/rfc3168.txt

[24] H. Schulzrinne, "Rtp : A transport protocol for real-time applications," RFC 1889, Janvier 1996. [En ligne]. Disponible : http://www.ietf.org/rfc/rfc1889.txt

[25] T. Friedman, R. Caceres, et A. Clark, "Rtp control protocol extended reports (rtcp xr)," RFC 3611, Novembre 2003. [En ligne]. Disponible : http://tools.ietf.org/rfc/rfc3611.txt

[26] M. Westerlund, C. Perkins, K. Carlberg, I. Johansson *et al.*, "Explicit congestion notification (ecn) for rtp over udp," RFC 6679, Août 2012. [En ligne]. Disponible : http://tools.ietf.org/html/rfc6679

[27] T. Moncaster, A. Jacquet, B. Briscoe, et A. Smith, "Re-ecn : Adding accountability for causing congestion to tcp/ip," Avril 2012. [En ligne]. Disponible : http://tools.ietf.org/id/draft-briscoe-conex-re-ecn-tcp-00.txt

[28] "Congestion exposure (conex). ietf working group." [En ligne]. Disponible : http://datatracker.ietf.org/wg/conex/charter.

[29] S. Krishnan, M. Kuehlewind, et C. Ucendo, "Ipv6 destination option for conex," Octobre 2011. [En ligne]. Disponible : http://tools.ietf.org/html/draft-ietf-conex-destopt-01

[30] V. Reguera, F. lvarez Paliza, W. Godoy Jr, et E. García Fernández, "On the impact of active queue management on voip quality of service," *Computer Communications*, vol. 31, no. 1, 73–87, 2008.

[31] M. Parris, K. Jeffay, et F. Smith, "Lightweight active router-queue management for multimedia networking," vol. 3654, 1999, 162–174.

[32] C. Hollot, V. Misra, D. Towsley, et W. Gong, "On designing improved controllers for aqm routers supporting tcp flows," vol. 3. IEEE, 2001, 1726–1734 vol. 3.

[33] S. Athuraliya, S. Low, V. Li, et Q. Yin, "Rem : Active queue management," *Network, IEEE*, vol. 15, no. 3, 48–53, 2001.

[34] S. Kunniyur et R. Srikant, "An adaptive virtual queue (avq) algorithm for active queue management," *Networking, IEEE/ACM Transactions on*, vol. 12, no. 2, 286–299, 2004.

[35] M. Fiedler, T. Hossfeld, et T.-G. Phuoc, "A generic quantitative relationship between quality of experience and quality of service," *Network, IEEE*, vol. 24, no. 2, 36–41, 2010.

[36] Q. Zizhi, S. Lingfen, N. Heilemann, et E. Ifeachor, "A new method for voip quality of service control use combined adaptive sender rate and priority marking," vol. 3, 1473–1477 Vol.3, Juin 2004.

[37] H. Sanneck, N. Le, M. Haardt, et W. Mohr, "Selective packet prioritization for wireless voice over ip," 2001.

[38] N. T. Moura, B. A. Vianna, C. V. N. Albuquerque, V. E. F. Rebello, et C. Boeres, "Mos-based rate adaption for voip sources," *Communications, 2007. ICC '07. IEEE International Conference on*, Juin 2007, 628–633.

[39] K. Fujimoto, S. Ata, et M. Murata, "Adaptive playout buffer algorithm for enhancing perceived quality of streaming applications," vol. 3. IEEE, 2002, 2451–2457 vol. 3.

[40] E. Jammeh, I. Mkwawa, A. Khan, M. Goudarzi, L. Sun, et E. Ifeachor, "Quality of experience (qoe) driven adaptation scheme for voice/video over ip," *Telecommunication Systems*, vol. 49, no. 1, 99–111, 2012.

[41] G. Rubino, "The psqa project." [En ligne]. Disponible : http://www.irisa.fr/armor/lesmembres/Rubino/myPages/psqa.html

[42] O. Hrvoje et Z. Drago, "Adapted e-model and dynamic adjustment of voip parameters," *Telecommunications (ConTEL), Proceedings of the 2011 11th International Conference on*, Juin 2011, 481 –486.

[43] "Network simulator. ns-3." [En ligne]. Disponible : http://www.nsnam.org/

[44] “wireshark.” [En ligne]. Disponible : http://www.wireshark.org/

[45] “Eclipse ide for c/c++ developers.” [En ligne]. Disponible : http://www.eclipse.org/downloads/moreinfo/c.php

[46] “Waf.” [En ligne]. Disponible : http://code.google.com/p/waf/

[47] S. Ramroop, “A diffserv model for the ns-3 simulator.” [En ligne]. Disponible : http://www.eng.uwi.tt/depts/elec/staff/rvadams/sramroop/index.htm

[48] H. Hassan, J. Garcia, et C. Bockstal, “Aggregate traffic models for voip applications,” *Digital Telecommunications„ 2006. ICDT’06. International Conference on.* IEEE, 2006, 70–70.

[49] K. Nichols et V. Jacobson, “Controlling queue delay,” *Communications of the ACM*, vol. 55, no. 7, 42–50, 2012.

[50] ETSI, “Quality of service (qos) measurement methodologies,” 102 024-5 v4. 1.1, Janvier 2003.